NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

CONSTRUCTING SOCIAL NETWORKS FROM SECONDARY STORAGE WITH BULK ANALYSIS TOOLS

by

Janina L. Green

June 2016

Thesis Co-Advisors: Michael R. McCarrin
Ralucca Gera

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704–0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave Blank)*	**2. REPORT DATE** June 2016	**3. REPORT TYPE AND DATES COVERED** Master's Thesis 01-01-2015 to 04-30-2016	
4. TITLE AND SUBTITLE CONSTRUCTING SOCIAL NETWORKS FROM SECONDARY STORAGE WITH BULK ANALYSIS TOOLS			**5. FUNDING NUMBERS**
6. AUTHOR(S) Janina L. Green			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Intelligence Agency			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: NPS.2012.0024-CR06-EP5-A; NPS.2015.0050-CR01-EP5-A.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT *(maximum 200 words)*

Intelligence analysts depend on the ability to understand the social networks of suspects and adversaries. We develop a novel method for automatically discovering this information from digital storage media by analyzing byte-offset proximity between digital artifacts on the raw media. We show that this method can be used to group email addresses that indicate real communication between users and those that do not. Furthermore, in the case where addresses do represent communication between users, our analysis indicates that classic measures of centrality are effective for identifying important nodes and close associates, and that further study of modularity classes may be a promising method of partitioning complex components. Finally, in support of the above work, we also created a tagged dataset of graphs for which ground truth was determined by interviews with the owners, and which can be used for future study in this area. Two objectives motivated this thesis, both of which serve the greater goal of making analysts more efficient. The first was to reduce the time digital analysts consume sorting through the results, in order to complete cases in a timely manner. The second was to eliminate data that was not relevant to discovering social networks, in order to achieve the ultimate goal of eventually paving the way for an automated process that identifies social structures.

14. SUBJECT TERMS social network analysis, social network structure, digital forensics, digital fingerprinting, bulk data analysis, visualization, email address analysis			**15. NUMBER OF PAGES** 101
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 **Standard Form 298 (Rev. 2–89)**
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

CONSTRUCTING SOCIAL NETWORKS FROM SECONDARY STORAGE WITH BULK ANALYSIS TOOLS

Janina L. Green
Captain, United States Army
B.A., Queens College, City University of New York, 2006

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

June 2016

Approved by: Michael R. McCarrin
Thesis Co-Advisor

Ralucca Gera
Thesis Co-Advisor

Peter Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Intelligence analysts depend on the ability to understand the social networks of suspects and adversaries. We develop a novel method for automatically discovering this information from digital storage media by analyzing byte-offset proximity between digital artifacts on the raw media. We show that this method can be used to group email addresses that indicate real communication between users and those that do not. Furthermore, in the case where addresses do represent communication between users, our analysis indicates that classic measures of centrality are effective for identifying important nodes and close associates, and that further study of modularity classes may be a promising method of partitioning complex components. Finally, in support of the above work, we also created a tagged dataset of graphs for which ground truth was determined by interviews with the owners, and which can be used for future study in this area. Two objectives motivated this thesis, both of which serve the greater goal of making analysts more efficient. The first was to reduce the time digital analysts consume sorting through the results, in order to complete cases in a timely manner. The second was to eliminate data that was not relevant to discovering social networks, in order to achieve the ultimate goal of eventually paving the way for an automated process that identifies social structures.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

API	Application Program Interface
DoD	Department of Defense
HDD	Hard Disk Drive
IRB	Institutional Review Board
NPS	Naval Postgraduate School
OS	Operating System
PDF	Portable Document Format
POP	Post Office Protocol
RAM	Random Access Memory
RAR	Roshal Archive
SSD	Solid State Drive
USG	United States Government

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I would like to thank my thesis advisers and mentors, Michael McCarrin and Dr. Ralucca Gera, for their guidance and patience throughout this process. I came to Professor McCarrin with an abstract idea that had no focus, and he helped me to shape it into a concrete and interesting topic. Professor McCarrin and Dr. Gera were both dedicated to my success providing an abundance of advice, support, and editorial review. They were extremely patient in regard to answering questions and providing resources to help with my thesis, in topics of digital forensics and graph theory, respectively. They are extremely talented, and I was very lucky to be their student. I want to thank you both for all your help and your flexibility over the past year.

I would like to thank all my NPS professors and classmates. I am grateful for the opportunity to have attended NPS and will always cherish the memories. My classmates were truly a team. The professors showed over and over again that they cared about students, dedicating additional time and resources to ensure their students thoroughly understood the topics.

I would like to thank my family, Ronald Green, Jada Green, and Jasmine Green, our newest member, born while working on this thesis. Ronald, I would not have been able to complete these two years without you taking such good care of our girls. Thank you for doing more than your fair share to allow me to concentrate on my studies. I want to also thank my mother, Evelyn Washington, who came down on numerous extended visits to help us with the children. Your love and support made this possible. Thank you, Mom.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1: Introduction

Even before the advent of commercial social networking sites, law enforcement and intelligence communities have been interested in social network discovery as a tool for understanding the hierarchy of entities they encounter in the course of their investigations. The ability of intelligence analysts to determine relationships between individuals is a powerful resource in discovering the structure of larger social organizations, such as corporations, cartels, cults, cliques and other groups connected by similar objectives or ideologies. Investigators use this information to identify people who hold leadership positions or possess key information as well as to analyze behavior patterns of individuals and groups.

Often investigators learn social network structures by questioning subjects during an interview or an interrogation. However, there are many problems with interrogating human beings, including reluctance of the interviewee to disclose valuable information, failure to remember information completely or correctly, or intentional misrepresentation of facts. Even when the interviewee is cooperative, the interview process is resource intensive in that it requires considerable analyst time and skill. Further, the interview must take into account the needs and rights of the interviewee—for example, an interrogation of human subjects involves far more restrictive time constraints than processes that deal with digital devices. Unlike the former, the latter can be copied reliably and in many cases stored indefinitely.

An alternative to interviewing is to find social networks by analyzing digital media owned or used by the subject of interest. Provided that this method can achieve results of similar quality, extracting social network information automatically from digital media offers analysts a way to obtain this information without the need for direct interrogations and their inherent limitations. As electronic devices become increasingly involved in all aspects of interpersonal communication, the strategy of extracting information about social network structures from digital devices becomes more and more promising. This holds true especially because the use of digital devices creates persistent digital artifacts: every time a person makes a phone call, sends a text, sends an email, or logs into a chat room she leaves a trace of her activity. Often, this trace is sufficient to allow an analyst to deduce the group of people she is in communication with and therefore is in some type of relationship with.

There are two methods currently available for accessing these traces. The first is through an interface into an information system that contains data related to social networking. Examples of such systems include dedicated social networking services such as Facebook and Twitter that offer users the ability to create profiles and curate a list of contacts. More generally, communication-related services like email servers, forums, chat rooms, etc., are also sources of social networking information. One strategy for obtaining a picture of a user's social network would be to access such services and analyze the contact lists they contain. The advantages of this method are easy access to the relevant data (provided we have the authorization) and the ability to quickly locate this data using the organization already defined by the system.

However, this method requires the analyst to overcome a number of challenges. Access may be restricted by permissions set by the interface. For example, if the target system is Facebook restrictions may be imposed by user settings or company policy and, much like in the interview process, these entities may have little motivation to cooperate with investigators. Users often have a high level of control over the data they are storing on online services, and can delete critical information. Moreover, information held by any single service may not be comprehensive for a given user, and the analyst may not have a complete list of all the services to which a user subscribes. The design of the interface may work against the goals of the examiner by failing to provide a mechanism for accessing the data of interest. For example, in the case where the target interface is a file system, artifacts stored in unallocated space are unavailable through the API. Finally, it is the difficulty of automating analysis, due to the dynamic nature of applications. Software tends to change over time and it is inefficient for engineers to constantly write new programs to read changing applications.

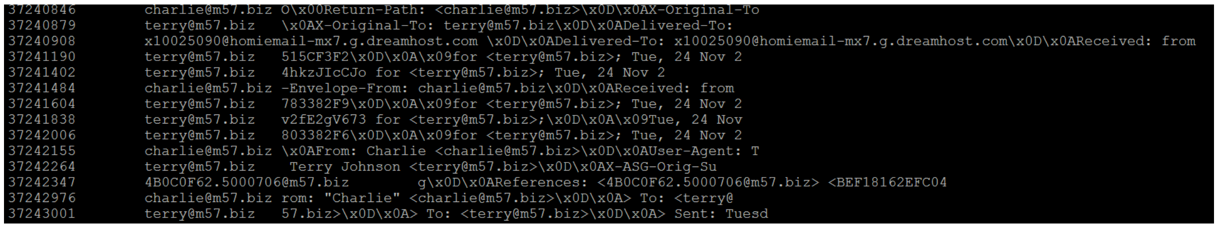
The second method is to bypass the system's interface and directly access the underlying data. This can be accomplished with digital forensic tools that read raw bytes from storage media. In our research, we explore this second option, since this method allows us to bypass the file system and restrictions placed by applications, conduct parallel queries, perform batch processing and access areas designated as “free space” or unallocated. This method does have challenges, which include acquisition of the media, the need to mine large volumes of data in limited time, the dynamic nature of the data, and the difficulty of visualizing the results. A detailed discussion of each of these issues follows.

The advent of cloud-based storage complicates data acquisition. Wireless capabilities have increased in availability and speed, causing data generated on digital devices to be more cloud based. The Chromebook, for example, pushes the user to store most of his data on Google’s servers rather than on his local machine. Cloud based storage has decreased the number of messages and records stored on local machines. These artifacts are instead stored in the cloud on private servers. These obstacles have increased the difficulty of discovering information about social circles on local devices, which in turn increases the difficulty of pin pointing possible witnesses and co-conspirators. Obtaining access to cloud-based data is hard. It is often difficult to get approval to access the servers this data is stored on, let alone approval to confiscate machines run by service providers. To request access to Cloud Servers, which may or may not be located in one’s home country, requires analyst time and resources and is not guaranteed to be successful. The goal of our approach is to find relevant information on local storage so that the need for further investigation of cloud-based storage is minimized.

Another obstacle in regards to data acquisition is data collection on local machines. Data acquisition is difficult because storage media is heterogeneous. Currently there are many different operating systems running on many different platforms. Any attempt to automate the collection process has to be able to consolidate information in a useful way from different data sets and different formats. Failing to solve this problem burdens the analyst with the task of sorting through many different types of systems, which only makes their job that much harder. A goal of the current work is to make the data collection and formatting automated so that analysts can dedicate more time to actual investigations.

The next problem is the volume of data. Traditionally, investigators use a myriad of techniques to collect digital evidence such as phone records, email messages, and text messages in order to discover social networks. Collection of this type of data has become increasingly tedious due to the decreasing cost of storage in digital devices and their increasing capability to store larger data sets. This creates a vast amount of data, which is difficult for investigators to sort through. Looking for relevant pieces of information through huge data sets is like digging for diamonds; an extraordinary amount of time and effort is consumed before anything of value turns up. As the amount of digital storage increases and its price decreases, the amount of data that an analyst has to study or look through becomes too much to process in a timely manner.

The last problem is the ability to display results in a format that humans can quickly process. One tool that we build on top of is Bulk Extractor [1] which is a digital forensic tool that extracts various data from storage devices. It is a widely used tool for extracting email addresses—both in law enforcement and other domains. One of the Bulk Extractors output is a text file with a list of email address, their offset on the storage device, and some content located around the email address. An example of its output is shown in Figure 1.1:



```
37240846        charlie@m57.biz O\x00Return-Path: <charlie@m57.biz>\x0D\x0AX-Original-To
37240879        terry@m57.biz   \x0AX-Original-To: terry@m57.biz\x0D\x0ADelivered-To:
37240908        x10025090@homiemail-mx7.g.dreamhost.com \x0D\x0ADelivered-To: x10025090@homiemail-mx7.g.dreamhost.com\x0D\x0AReceived: from
37241190        terry@m57.biz   515CF3F2\x0D\x0A\x09for <terry@m57.biz>; Tue, 24 Nov 2
37241402        terry@m57.biz   4hkzJIcCJo for <terry@m57.biz>; Tue, 24 Nov 2
37241484        charlie@m57.biz -Envelope-From: charlie@m57.biz\x0D\x0AReceived: from
37241604        terry@m57.biz   783382F9\x0D\x0A\x09for <terry@m57.biz>; Tue, 24 Nov 2
37241838        terry@m57.biz   v2fE2gV673 for <terry@m57.biz>;\x0D\x0A\x09Tue, 24 Nov
37242006        terry@m57.biz   803382F6\x0D\x0A\x09for <terry@m57.biz>; Tue, 24 Nov 2
37242155        charlie@m57.biz \x0AFrom: Charlie <charlie@m57.biz>\x0D\x0AUser-Agent: T
37242264        terry@m57.biz    Terry Johnson <terry@m57.biz>\x0D\x0AX-ASG-Orig-Su
37242347        4B0C0F62.5000706@m57.biz        g\x0D\x0AReferences: <4B0C0F62.5000706@m57.biz> <BEF18162EFC04
37242976        charlie@m57.biz rom: "Charlie" <charlie@m57.biz>\x0D\x0A> To: <terry@
37243001        terry@m57.biz   57.biz>\x0D\x0A> To: <terry@m57.biz>\x0D\x0A> Sent: Tuesd
```

Figure 1.1: Bulk Extractor email.txt output from a M57-Patent Drive, Realistic Data Corpus

After using the software on our test set, this file would tend to have hundreds of thousands of lines of output. It is not easy to determine what is important from hundreds of thousands of lines of text. Our goal was to take that information and process it to show the most relevant data. It is important that all the essential data be captured while not overwhelming analyst with busy screens. The simplicity desired needs to be balanced with ensuring that critical information is not left out. In addition, background information that makes a scenario clearer must be readily available when requested, without cluttering up the screen when it is not. Currently, there is nothing on the market that offers such a service, which created a void that could be fulfilled by our research. There is a need for automated systems that enhance the ability of investigators to perform analysis quickly and effectively. An analyst uses adaptability from experience and expertise to come up with the correct solutions to solve a case. It is a person's ability to process human behavior that would make them better than a computer at recognizing true patterns of social networks. Therefore, a solution is needed that couples a computer's ability to automate tedious tasks and people's ability to use their best judgment.

1.1 Motivation

Two objectives motivated this thesis. The first was to reduce the time digital analysts consume sorting through the results, in order to complete cases in a timely manner. The second was to eliminate data that was not relevant to discovering social networks, in order to achieve the ultimate goal of eventually paving the way for an automated process that identifies social structures. Both these objectives are intended to make analysts more efficient during their investigation.

In light of these objectives, this thesis seeks to address the following research questions:

- Can the proximity and quantity of co-located email address pairs on secondary storage media be used to discover information about social networks?
- Can these properties be used to distinguish between true positives (email addresses used for communication between human users) and false positives such as text which happens to have the commercial at symbol.

Analysts can currently extract email addresses from digital media using Bulk Extractor. However, Bulk Extractor's email scanner has a high false positive rate. This means that analysts have to manually sort through many results that do not indicate evidence of real communication between human users. The current research attempted to make strides towards methods to reduce false positives, and identify potential associates. Our goal was to make progress towards a tool that would automate social network discovery by focusing on email addresses found in digital media.

A secondary goal was to create an automated process to distinguish email addresses that do and do not provide useful information to the analyst.

1.2 Contributions

We develop a new method of analyzing relationships between email addresses on drives which relies solely on byte-offset proximity between the addresses and is therefore independent of OS, file system, or application-layer structures. By creating graphs of email addresses where linked nodes represent email addresses co-located in the same area of a storage device, we are able to quickly create graphs where components consist of related email addresses. We show that this method can be used to group email addresses that indicate real communication between users and those that do not. Furthermore, in the case where addresses do represent communication between users, we demonstrate that the resulting components contain information about the users' social networks.

By visualizing the graphs produced in this analysis, we develop a novel method of analyzing forensic artifacts, which has the potential to reduce analyst workload. In addition, we perform a preliminary analysis of the properties of our two types of components (those that are not related to social network information and those that are). With respect to graphs that do not display social networks we conducted cross-drive analysis and recorded which graphs were similar to graphs on different storage devices in our dataset. For graphs that do display social networks, we identify four major classes of graphs that merit further study. Furthermore, we record node count, edge count, modularity, average weight, eccentricity, and closeness scores, as well as which type of centrality best displayed the drive owner's closer associates. The two measures tested were Eigenvector and Betweeness. Our early findings suggest that future research into using these metrics to automatically classify email addresses is promising.

In support of the above work, we also created a test set of drives for which ground truth was determined by interviews with the owners. We tagged these graphs based on whether or not the graphs demonstrated social networks or if the results were unclear. Graphs that were tagged as not containing social network information can be used by those interested in future work to identify and remove graphs that do not display social networks. Additionally, graphs tagged as interesting can be used by those who would like to prototype algorithms for identifying social network graphs or identifying graphs that do not display properties of social network graphs.

1.3 Thesis Structure

The thesis is organized as follows. Chapter 2 presents the background information on forensic and visualization techniques, as well as previous work conducted in this area. Chapter 3 describes the tools designed for our social network graphs. Chapter 4 presents our methodology and experiment design for creating social network graphs that display syndicate organizations. Chapter 5 describes our experiments and analysis. In Chapter 6 we present our conclusion and propose areas for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Technical Background

This chapter presents higher level concepts essential to our research. We begin with an introduction to Network Science, which includes a series of useful definitions. Afterwards we discuss properties of secondary storage and bulk data analysis. These concepts are important because in our research we created network graphs by extracting data using bulk data analysis tools. We also analyzed them using Network Science concepts. The data we looked for were email addresses located on storage devices. Therefore it is important to have a basic understanding of secondary storage devices, email protocols and formats.

2.1 Network Science Concepts

Network Science is an area of study that focuses on the interconnection of things. The things can be in any field from the relationships of routers in computer networks to actors playing in the same Hollywood movies. The properties of the relationships provide interesting and useful insights, such as which animal is at the top of the food chain, how many airplane routes can get you from New York to San Francisco, or what group of people are likely to study for a midterm during the Super Bowl.

In order to study these connections we create networks. A network is a derivative of a graph, which will be explained in further detail in 2.1.1. A graph is used in Graph Theory to study the mathematical properties of nodes and edges and the relationships between them. A network is used to study the relationships between discrete objects. Often network and graph is used interchangeably. We felt it was important to highlight the subtle difference and to stress that Network Science was developed from Graph Theory.

Sometimes, learning about a particular network is not enough. You might want to compare different networks to each other. Reasons for this would include determining which is more efficient. For example, in a network of highways it might be better for traffic if they all connect at a common location or it might be better if they do not connect at all. You would have to compare the two. Before we continue further, below is a section on common terminology to ensure our usage of these words is clear.

2.1.1 Terminology

First we give the definitions of networks and graphs. Then we provide the definitions of the components that make up a network including edges, nodes, and components. We will then review ways to classify the parts of the network, and finally ways to classify the entire network itself.

Basic Concepts

Graphs, Sub-graphs, and Social Networks: As previously stated a graph is an interconnected group of things and a network is a graph that has concrete properties. The term graph is often used in place of the word network. The definition is the following: "A *Graph G* is an ordered pair of finite disjoint sets (N, E) such that E is a subset of the set $N \times N$ of unordered pairs of N. The set N is the set of *[nodes]* and E is the set of *edges*. If G is a graph, then $N = N(G)$ is the node set of G, and $E = E(G)$ is the edge set. An edge {x, y} is said to *join* the nodes x and y and is denoted by xy. Thus xy and yx means exactly the same edge; the nodes x and y are the *endnodes* of this edge" [2].

As the complexity of the network increases, it becomes increasingly difficult to understand. We want to break the graphs down into manageable clusters of related items to study. These are called sub-graphs since they are smaller components of the larger network. A *Sub-graph* "is a subset of nodes and edges within a larger graph." [3].

The networks we discuss involve people. When the relationships in the network are between people these networks are called "Social Networks". "A *social network* is a social structure made up of individuals (or organizations) called 'nodes', which are tied (connected) by one or more specific types of interdependency, such as friendship, kinship, common interest, financial exchange, dislike, sexual relationships, or relationships of beliefs, knowledge or prestige." [4]

Network Components

Nodes, Edges, and Components: A network contains the items that we are studying. In this paper we will refer to those items as nodes. In Network Science or Graph Theory materials, you will often hear them called actors or vertices as well. "A [node] is a discrete individual, organization, event or collective social entity that links to others in a network." [5]

The connections between nodes are called *Edges.* These edges represent relationships between nodes, an individual, a pair or a group. In a network of police officers, you can have edges which represent who an officer went to respond to an incident with. If in the graph you allowed edges to self-loop then you could tell who responded to the most calls by the amount of edges a node has to itself. You may also be able to determine who a particular officer's partner is, since partners tend to respond to incidents together. Edges are also referred to as ties or links.

A graph can have one or more *Components.* A component of a graph is a set of nodes that are connected by some path. For example, let us consider a classroom of elementary students, where the students are represented by nodes and the relationship is between nodes that have the same teacher. If the graph included one class then all students would be connected and the graph would have one component. If the graph included all classes in the elementary school then there would be as many component as there was classes, assuming a teacher did not teach more than one class.

Characterizing Nodes

Clustering Coefficient, Centrality, Closeness and Degree: When classifying a network, its nodes, edges and components can be characterized. In our research we focused on the statistics surrounding the various nodes. In particular we were concerned with the clustering coefficients, centrality, closeness and degree. The "*Clustering Coefficient* is a measure of the likelihood that two associates of a node are associates themselves. A higher clustering coefficient indicates a greater 'cliquishness'." [4]

The formula is the following [6]:

$$\bar{C} = \frac{1}{n}\sum_{i=1}^{n} C_i \tag{2.1}$$

where C_i is the coefficient for every node, n is the total number of nodes in the graph and $\bar{C}$ is the average coefficient.

The "*Centralization* of a network is the extent to which a network is centralized around one or a few central [nodes]." [7] The formula is the following [8]:

$$\frac{\sum[c_* - c_i]}{max\sum[c_* - c_i]'} \tag{2.2}$$

where c_i is the centrality of node i and c_* is the centrality of the most central node."

The "*Centrality* of [a node] is the extent to which [a node] occupies a central position in the network in one of the following ways: having many [edges] to other actors(degree centrality), being able to reach many other [nodes](close centrality), connecting other [nodes] who have no direct connections (betweenness centrality), or having connections to centrally located [nodes](eigenvector centrality)." [5]. The methods of centrality that we will focus on is degree, betweenness, and closeness.

The "*Betweenness Centrality* of [a node] is the extent to which [a node] serves as a potential 'go-between' for other pairs of [nodes] in the network by occupying an intermediary position on the shortest path connecting other [nodes]." [5] The formula is the following [3]:

$$B(N_i) = \sum_{j<k} \frac{S_{jk}(N_i)}{S_{jk}} \tag{2.3}$$

where $B(N_i)$ is the betweenness of node i, S_{jk} is the number of shortest paths between nodes j and k, and $S_{jk}(N_i)$ is the shortest path between j and k that contain node i.

The "*Closeness Centrality* of [a node] is the extent to which the most direct path connecting [a node] to each of the other [nodes] in a network are short rather than long. This measure is only meaningful for a fully-connected network in which there are no isolated [nodes]. A high closeness score means [a node] can access many other [nodes] and is therefore relatively independent of the control of others." [5]

The formula is the following [3]:

$$C_C(N_i) = \frac{1}{[\sum_{j=1}^{g} d(N_i, N_j)]}(1 \neq j) \tag{2.4}$$

where C_C, the Closeness Centrality of node i is the inverse of the sum of the shortest path distances connecting every pair of nodes. The degree of each node depends on how many different nodes are connected to it.

The "*Degree Centrality* is the number of connections that [a node] has in a network." [5]

The formula is the following [3]:

$$C_D(N_i) = \sum_{j=1}^{g} X_{ij}(i \neq j) \tag{2.5}$$

where $C_D(N_i)$ denotes degree centrality for node i and $\sum_{j=1}^{g}$ counts the number of direct [edges] that node i has to the other j nodes in the graph.

Characterizing the Network

A goal during our study, was to calculate normalized statics so that we can compare graphs of different shapes and sizes. Typically, we were interested in the connectedness, size, and density in a graph. Many times these calculations have predictive powers. For example, if you are looking at a communications network and track how information is passed through the nodes you can predict the minimum number of nodes needed to spread information as quickly as possible to everyone in the graph. In fact many businesses do this type of analysis in order to conduct targeted advertising.

"A *Connected Network* is one in which every [node] can reach every other [node] either directly or through an intermediary: there are no isolates." [9]

The size of the network depends on how many items (nodes) are in it. For example, a graph of a standard elementary school class would be around 20-30 nodes. The density is dependent on how many edges a graph has running through it, the more edges, the more dense it is. At times a network displays special properties such as having a community or a clique. A community in a graph, is when a group of nodes have more connections(increased density) within a group then to other nodes in the graph. "*Communities* are a sub-graph of a network whose nodes are more tightly connected with each other than with nodes outside the sub-graph." [10]. A clique is where all nodes in a group have connections to each other. More formally, "A clique is a group of three or more nodes that are all are connected to each other and have no common edge to any outside nodes." [5]. In a clique everyone knows everyone, so there is an edge between any two nodes.

"A *k-Core* " is a component of a graph that is created by removing all nodes which have a degree less than k. The definition is the following: "Let G be a graph. If H is a subgraph of G, $\delta(H)$ will denote the minimum degree of H; each point of H is thus adjacent to at least $\delta(H)$ other points of H. If H is a maximal connected (induced) subgraph of G with $\delta(H) > k$, we say that H is a k-core of G [11]."

The *k-Core* with the maximum value of k that yields a non-empty graph is referred to simply as the *core*.

2.2 Properties of Storage Devices

We obtained storage devices to study the information they had written on them. It is valuable to understand how they work. Hard drives are one type of digital storage device. Hard drives have a series of platters that can store digital information. The platters are separated into sections called tracks and sectors. These sections are similar to file cabinets with a multitude of draws and folders holding important documents.

The goal for storage devices is to allow users to access information quickly. Offices tend to achieve this goal by organizing their paperwork in file cabinets and operating systems similarly have a filing system to organize data on the hard drive. Documents that are related are usually stored in the same area in an office filing system and as such documents that are related are for the most part stored in the same area on the hard-drive.

While flash memory does not have moving parts like hard drives, they still hold information by partitioning the memory into files and folders. Related information tends to be stored together. This is important because in theory we can grab a chunk of data or set of folders to continue with the office analogy, and it should contain information that is related to each other. Operating Systems do not always organize data in close proximity but for a majority of similar files they do and it is this spatial locality that we attempt to take advantage of with our research.

We gain access to storage devices in order to exam them, by collecting disk images. A disk image is an exact copy of an entire storage device. Not only does it copy the contents but the structure of the storage device also. Normally, a person may just copy files off of another person's storage device, for example all the pictures from last summer's vacation. However, sometimes much more is needed. Maybe the computer has a lifetime of special software and a fancy operating system that should be retained? In this case a disc image is needed to copy the entire structure and contents of the computer the way they are.

2.3 Email Protocols and Formats

It is important to understand email protocols and formats. In our research we studied email addresses that were extracted from storage devices. Understanding the properties of these protocols or formats is essential to understanding why we expected to see a relationship between email addresses.

Personal Storage Table (.pst) and Off-line Storage Table (.ost) are file formats created by Microsoft for use with Microsoft Exchange, Windows Messaging and Microsoft Outlook. These formats allow Outlook users to store their emails on their device directly or on a secondary storage system. These file formats are a fixed size and can increase when additional data is stored. When data is erased they are not reduced in size. Instead the space is marked as free and will remain until overwritten by some new data. These formats appeal to users because when they store their emails locally, they do not use up their allocated server space and they have access to their data when they are off line.

MBOX is a name for various file formats to include mboxo and mboxrd. These file formats hold all email messages in a single file. All the messages are appended to each other. The format knows where an email starts with the word “From” or whatever word is typically listed first in the email client being used. EML(.eml) is a file format typically used by Outlook Express. It is used to save or send an email. It stores the email message and any attachments or links.

Personal Address Book(PAB) and Offline Address Book (OAB) allow users to have access to their contacts when they are not connected to the Internet. When the user is online, their address book is updated and when they go off-line their contacts are stored on their local machine. A VCARD is a virtual business card. This means it has all the information you would typically find on a business card, but instead of being physical it is digital. VCARDS are usually sent as an attachment to an email message.

Post Office Protocol(POP), POP2, and POP3 are email protocols that store emails on a remote server until a user downloads that email to their local machines. Once that happens the email is deleted from the remote server and only stored locally. Internet Message Access Protocol (IMAP) is a protocol that stores emails on a remote server until the user deletes the email from the server. Message Application Programming Interface (MAPI) is Microsoft’s proprietary email protocol and is similar to IMAP in that it allows emails to be stored on their remote servers until deleted by a user.

Regardless of which protocol is used, there are email clients such as Thunderbird that allow users to store a copy of their email messages on their local machine. Email clients are different from web based email because with web based email a user would have to connect to the Internet to create new messages and access stored messages. Email clients are stored on a user’s local machine. While a user needs to be online to send and receive mail, with an email client they can read archived messages and create new messages.

Many users have free web based email servers such as Gmail, Hotmail and Yahoo. These web based email servers make it difficult to recover email from local hard drives since they are stored remotely. However, when users pull and read messages from off their mail severs these items are cached and can be recovered. In addition, many local machines save frequent user name and passwords to make it easier for users to log into their various online accounts. These bits of information can be useful for investigators to discover email addresses used for log-ins and email addresses of frequently contacted individuals.

2.4 Forensic Analysis Tools

As mentioned in Chapter 1, we had to overcome the fact that devices are managed by different file systems. A file system is the data structure that an operating system uses to access its files. Imagine trying to use the Army's filing system, the Modern Army Record Keeping System to access files in libraries, which use the Dewey Decibel Classification System. This problem made it challenging to use one application to navigate different file systems.

2.4.1 Using the System Interface

Despite the inherent challenges of creating general solutions for heterogeneous interfaces, such as file systems, there were times when it was useful to inspect metadata related to the artifacts we were examining, such as the file path, if any, associated with the sectors from which the addresses had been extracted. The file path and file name offered clues as to the origin of the data. When we wanted to further investigate data from the image, we attempted to use The Sleuth Kit to navigate the file system. The Sleuth Kit is a suite of command line tools designed for forensic analysis of disk images.

The Sleuth Kit is notable because it *does* attempt to understand all major file systems. It is relatively successful since it is maintained by a large community and kept up to date. However, it has its limitations, which are discussed in the introduction. For the majority of drives The Sleuth Kit was a useful tool but for two drives the tool failed.

2.4.2 Bypassing the System Interface (Bulk Data Analysis)

Because it is not feasible to write a program for every different type of file system, we choose to use as our primary tool a program that bypassed the file system altogether. We had access to a program that was able to read data from storage devices directly.

Not only was this beneficial because it allowed us to study devices with different file formats, but it allowed us to obtain data from unallocated space without using additional tools. Unallocated space is the free space on a storage device. Many times a user will write data to a storage device and delete that data. That data is not labeled as unallocated since it can be used for new data. However, if the user has not written new data to that space, it is still there until overwritten by new data.

The program we used to extract data is called Bulk Extractor. Bulk Extractor is an open source tool developed at the Naval Postgraduate School that extracts a plethora of information from digital storage devices. The program reads a hard drive from the beginning to the end and looks for data that matches different criteria. The developers call the functions that look for different characteristics scanners. The scanner that we used was the email scanner that pulls anything that looks like an email address. The program can be run on digital media directly or images of file systems. Bulk Extractor can extract email addresses even if they are contained in compressed files such as ZIP or PDF files. It can also extract email addressed from encrypted RAR files.

Bulk Extractor has a histogram feature. The histogram feature shows the most common email addresses. This is an important feature since the most common email address can be used to infer who may own the device or what kinds of programs are installed. When Bulk extractor finds email addresses it discovers on a drive, it also records their address(offset) on the hard-drive. It does this from the beginning to the end of a storage device and writes its findings in a list format to a text file. The list includes three columns which are offset, email-address, and a small amount of data content surrounding the email-address. This is displayed in Figure 1.1.

While Bulk Extractor does a great job in automating extractions of email addresses there is still an issue of getting a list with too many irrelevant email addresses. An example is a generic email used by an application such as helpdesk@thisapplication.com. These types of email addresses may not indicate any relationships and instead are a common email type found on any device hosting that application.

While all false positives cannot be suppressed, Bulk Extractor attempts to reduce the number of irrelevant email addresses. Bulk Extractor conducts the suppression with the list of email addresses that should be disregarded. The Bulk Extractor manual calls these lists stop lists. A stop list is a list of items that we want excluded from being searched for on the hard drive. As these email addresses are being discovered, users can add email addresses to the stop lists. The stop list stores the email-address and the small amount of text that is located directly before and after the email's address in memory.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3: Related Works

The goal of our research was to ease the burden on forensic analysts, by developing an automated means to sort through digital storage and extract social networks. We studied scholarly papers that discussed methods, software, and research that could be used to make our program efficient and avoid mistakes by reviewing lessons learned. This section is a brief summary of papers that were related to our work and some ideas that were incorporated. The papers discuss the problems in digital forensics that created the need for the tool we built and concepts that were important to creating the right tool. These concepts include visualization techniques, social network discovery, email analysis, filtering irrelevant data, and cross drive analysis.

3.1 Forensic Tools Are Becoming Obsolete

Garfinkel [12], argues that in the past digital forensics was an important asset that helped investigators solve cases quickly. However, the benefits that digital forensics are able to offer are diminishing as other areas in technology advance. Not only is the immense amount of digital storage a road block, but we also have various amounts of file formats, encryption, and lack of training on programs that assist with analyzing data. Garfinkel's paper explains that without an increase in standardized research, the benefits of digital forensics will be lost.

A consolidated effort from different organizations is promoted in his paper, so research can be moved forward by better communication, documentation and teamwork. Instead of building forensic programs from the ground up, researchers would rather improve upon programs already available. For an analyst it would be useful to have a set of standards and rules, so that forensic programs can be compatible with each other and an analyst doesn't need to keep databases of separate results. Building various programs from scratch is not as efficient as communities working collectively to improve the efficiency and functionality of one program. Common file formats and modules that build upon each other will ensure movement forward in research. That is why Bulk Extractor is important as an initial tool for our methods. It works on any storage device and does not depend on a particular operating system. It is a universal tool that can be used across platforms.

Garfinkel [13] points out a few problems with present day forensic tools. Tools are designed to find specific information and not to assist in overall investigations. For example, a tool may extract social security numbers but investigators may need to know which exact social security number has been tied to numerous transactions. Tools look for information where evidence resides on the computer system and not where crimes were committed with computers or against computers. For example, if a person was accused of credit card fraud, a tool may try to find evidence of credit card numbers; instead of determining if the computer's IP was used in the online credit card transaction or used to remote into another system. While any given system may not contain evidence directly they can offer clues to assist with finding the actual evidence or insight to what issues are being analyzed, and what tools should assist with this process in a holistic manner. We want to build a program that incorporates a holistic methodology. We do not want investigators to extract our data and have to provide specific email addresses they are looking for. We want the program to provide clues and highlight relationships for them.

Another issue pointed out by Garfinkel is that tools on the market today are attempting to provide complete results with no regard to speed so they are not timely. This means that if analysts need to obtain information quickly they may not be able to. The last issue addressed by Garfinkel is that there is not enough on-the-job training of digital forensic tools.

This thesis has taken these issues into consideration. This research addresses them by using open-source tools, methods that can be performed on any operating system, and by using holistic output that provides insight into data and does not just produce output based on specific search queries. The processes described in this thesis outputs information about social networks that can be useful in analysis. The information obtained can be saved and used in other investigations.

3.2 Visualization

Due to our ability to process visual information better than reading text, data displays help analysts reason about data, see patterns, relationships and correlations.

Garfinkel [14] discusses how visualizations for forensic discoveries make patterns more obvious and help to detect clusters and outliers. He suggests that the visualizations should be created automatically using code, that each graph or chart should be unique based on the data input. This is opposed to those created by human design, where the input is the same but the output is unique based on the designers' preferences. When a program runs with the same inputs, analysts should obtain the same exact display, and not graphs that look completely different. The is important to maintain integrity of reports and to increase the speed at which information is being processed and used by the analyst.

Marty [15] also discusses how visual analysis can be useful to unlock insight. The author discusses that there is an optimal mix between human reasoning and data mining. His paper shows that visual analytics are a promising method to help analysts capture relevant knowledge from data exploration and thus find this optimal mix. He warns, however, that visualization is not a panacea but that algorithms play a vital role to support relevant data. Algorithms should help to reduce the volume of data and highlight the most important information. Some algorithms include the use of non-hierarchical and hierarchical visualization techniques. For non-hierarchical approach, the author uses size and color to show the attributes of file size, creation time, access time and modification time. For the hierarchical approach, the author uses graphs and tree maps to demonstrate relationships and organizational structures. Evaluation results demonstrate that visuals can increase the speed and accuracy in finding valuable tidbits in a haystack of information.

Teerlink and Erbacher [16] discuss hierarchical and non-hierarchical visualization techniques to reduce the amount of time needed to pick out "interesting" data. Using visuals was an effective way for analysts to discover relevant information faster and collect more relevant data. Teerlink and Erbacher state that on average 53% more files were located using their techniques and that there was a 37% percent reduction in time to locate files. In addition, their results suggest that this is an easier method because analysts found the first file 57% percent faster. These results were in comparison to using the Linux command line to find files.

Shneiderman [17], discusses a tree-map method for visually displaying the size of files on a hard drive. The paper explains that using $2D$ visualizations are easier to view, and they are less data intensive than $3D$ visuals. The problem Shneidermann encountered was limiting the size of the trees to be displayed on a screen. He suggests that having to scroll through too many pages or making the trees small took away from the benefits from being able to show this information visually. He plays with different methods to make the visual better for users in terms of speed and display. One can certainly relate to this by doing a Google search: if the returned pages would not be ordered based on their relevance, we would have to scroll through pages of website and check each link. Optimization was achieved by manipulating alternative layouts, copying files, trying different color displays, and calculating different functions performed on files.

3.3 Social Networks

Rowe et al. [18] explain techniques used to discover social networks through email. One of them includes algorithms to find sub-graphs. The sub-graphs determine what people belong to what social groups or cliques. Determining the number of cliques a person is a member of, is one criterion to specify if they hold a higher position on the social ladder. In addition, being a part of a larger clique also increases the person's social ladder score. Email and average response time determines what cliques people are a part of and how important they are. This thesis only deals with analysis of local drives, but Rowe's paper also has methods for cross-drive analysis. The cross-drive analysis methods are explained and suggested in chapter 6, for those thinking of continuing this research.

Campbell et al. [19] detected communities by high connectivity within a group and low connectivity outside a group. They found connections to be high within communities and low across communities. They found that these communities corresponded to organizations or groups on real social networks. We attempted to mirror this technique with determining edges by the amount of emails sent between nodes.

Bird et al. [20] raised an interesting problem, which we also encountered in reference to email alias. It is not unusual for one person to have many email addresses and user names. It is important when developing these graphs to not mistakenly count one person as many people due to confusion between email addresses and activities. The same person using different email addresses could be doing so purposely to separate their communication between various communities. An example, would be keeping church organization communications completely separate from fraternity organization communications. This would be useful information to know.

The authors developed an algorithm to handle the cases of email alias. They found that aliases had similarities. For example, it would include parts of the first and last name. So their algorithm compared email addresses and gave them different scores. If the scores were close enough they considered them a match and merged the email addresses into one. They first had to normalize the email addresses and user names by getting rid of capitalization, punctuation, and generic terms such as "admin" or "support". After that they proceeded to give two scores based on name similarity. They based similarity on the Levenshtein edit distance, and if it had the initial of the first name and complete last name or complete first name and last name initial. They then added these scores together and made a determination. The Levenshtein distance is a string metrics that measures the difference between two strings based on how many edits would be necessary to change one word into the other. The last piece of interesting information discovered by reading these papers was that they both displayed the small world concept: while every person did not email every other person, once you built the graph you could see a path established to everyone. So you could reach every person in the graph through the network of users.

3.4 Email Analysis

Boucher and Kuange [21] discusses how emails are increasingly being used to obtain information. This is because the informality and immediate response associated with e-mail communication, causes people to let their guard down. They do not tend to think about the fact that an e-mail, even deleted e-mails can be on their computer system or on servers still intact for forensic analysis. The article also discussed different commonly used email systems. The email system that was most interesting to this work was the email client system. This type of system uses applications stored on a user's computer system. Emails are downloaded and stored on the user's personal computer. Examples given were Microsoft Office Outlook, Mozilla's Thunderbird, Windows Live Mail, Outlook Express and Eudora.

3.5 Getting Rid of Junk

As pointed out by Rowe et al. [18] many items on a hard drive do not provide forensically useful information about users. These are usually operating system and application software related information. It is important to get rid of this irrelevant data because digital forensics has resulted in increasingly larger data sets. We need to get rid of the uninteresting [6] so that we can focus our attention to items that will help in an investigation. Rowe [6] eliminates uninteresting files by comparing hashes to known uninteresting files. We attempt a similar technique by comparing uninteresting email addresses with known email addresses that are used for applications and operating systems. These known email addresses come from the software Bulk Extractor's stop list. In addition, we were able to add additional email addresses to the stop list from work being done by Rowe et al. [22] to discover uninteresting email addresses. In our thesis we used the stop list from Bulk Extractor to weed out email addresses that were not of high interest. These includes email addresses that were on every system because they belonged to root certificates, operating system developers, or other common application software installed on most personal computing devices.

3.6 Cross Drive Analysis

Garfinkel [14] states, "Today's forensic examiners have become the victims of their own success." This is because there are more digital drives being obtained than there are examiners or resources available to analyze the data. Examiners are currently [14] making copies of these drives and conducting searches of each drive independently. Garfinkel's [14] discusses several problems with this technique. They are improper prioritization, lost opportunity for data correlation, and improper emphasis on document recovery. In a perfect world, examiners would be able to go directly to the areas on the disc that are relevant to their investigation. However, since that is not the case we talked about automation and visualization to help to help lighten the load. The previous techniques discussed will help alleviate the problem of improper prioritization. The automation is used to pull out the data that investigators are looking for and the visualization will display it in a way that investigators can make sense of.

There is still a huge issue of time management. While these techniques save investigators time there is still a limit to how fast computer systems can process data. While the plethora of confiscated drives continues to increase, computer systems may take hours to image and pull valid information from each individual drive. This thesis deals with the time management of the investigator and how we can use the technology we have today to leverage that time. Breakthroughs in future technology will no doubt increase the gains presented in this thesis, when computer systems are able to process work at a faster speed.

Another problem discussed by Garfinkel is lost opportunities, this is important and something within reach. He points out that "Because each drive is examined independently, there is no opportunity to automatically "connect the dots". The responsibility is on the analyst to recognize these correlations. In this thesis, we will test our methods on individual hard drives. Experiments related to connecting correlations across drives will be discussed in 6.2, the section on future work. Garfinkel has a simple statistical technique to identify drives that hold more forensic value and should be given a higher priority. He suggested downloading numerous images to a corpus and analyzing them in unison rather than independently. This advice will be useful for cross drive analysis work. The last problem discussed is emphasis on document recovery. Garfinkel states that with the current forensics tools available on the market the focus is on recovering lost or deleted data. He suggest that this reasoning is flawed and that the focus should be on obtaining data that is going to further an investigation, not on every piece of data that is deleted or hidden. We took this advice and built our methods on the philosophy of gaining information of quality (interesting email addresses that would be relevant to social connections) and not just quantity of email addresses extracted.

These studies highlighted the problems that we needed to focus on, when we developed our tool for extracting social networks. The problems that we considered when we developed our tool were: increasing storage sizes, various file formats, visualization burdens, removing irrelevant data, and cross drive analysis. Methods discussed in the studies that were incorporated were visualization methods, social network techniques, and email analysis. None of these studies attempted to create the exact tool we built or create a method to specifically extracts email addresses based on byte-offset and frequency between email address pairs. These papers were all useful to consider methods, best practices, and concerns to be dealt with before we created our tool for social network extraction.

CHAPTER 4: Methodology

Our primary experiment was intended to determine whether our technique of constructing graphs using byte-offset proximity between email addresses found on the raw media is sufficient to reveal correlations between the addresses that are of potential use in identifying and then analyzing the social network of the device owner. Answering this question required considerable preliminary preparation: before we could test our method we first had to generate a dataset where ground truth was known—that is, where extracted email addresses could be reliably tagged as related to social networks of the device owner or not. A brief outline of the steps we took to obtain such a dataset and test our method against it follows:

1. Recruit volunteer device owners, collect their test disk images and produce graphs.
2. Separate graphs into three bins: Useful, Not Useful, or Unclear, where "usefulness" indicates the presence of social-networking information.
3. Analyze the components in the various bins.
4. Conduct exit interviews with the owners to have them confirm or refute results of our analysis.
5. Inspect graphs with unclear results and conduct further analysis.

4.1 Graph Creation

In order to create graphs of social networks a number of initial steps had to be conducted:

1. Obtain an image of a storage device.
2. Extract the email addresses and the byte-offset of their location on the storage device, relative to the first addressable byte on the device.
3. Construct graphs from email addresses using the addresses as nodes and placing edges between addresses that appear within a fixed byte window. For each graph, extract the twenty largest components.

4. Examine the results in Gephi.

4.1.1 Obtaining Images for Analysis

We were fortunate to have three data sets to work with: the Realistic Data Corpus drives, the Real Data Corpus, and Volunteer Drives. The images from the Realistic Data Corpus and the Real Data Corpus were already stored on servers located at the Naval Postgraduate School. The images from the Volunteer drives had to be collected. These drives provided us with a real world test set with known ground truth.

Realistic Data Corpus

The Realistic Data Corpus is a dataset of storage media images created by different groups of Naval Postgraduate Computer Science Interns [23]. This dataset was created for digital forensic education. Some of the images were created by simply storing a few pictures on the device and then deleting them to demonstrate that deleted data can be restored and others were more elaborate scenarios of crimes being committed. Scenarios included a corporation holding patents, an organization selling weapons and the last was an organization trafficking drugs. The interns had to plant documents, use programs, and conduct Internet searches that would support these roles. For example images of "cats" could be considered illegal in the scenario. The Realistic Data Corpus was a good test set because we knew what the scenarios were and could determine if any of the characters and their accomplices were shown on our graphs. A problem with this dataset, however, was that it was not sufficiently complex and it was small.

Real Data Corpus

The Real Data Corpus consists of disc images collected and maintained by the Naval Postgraduate School. The devices from which the images were taken include hard drives, thumb drives, SD cards, and compact discs, purchased resale from thirty different countries outside the US. This data set contains $3,238$ different drive images, all of which were previously owned. These images contain data from their previous users for two reasons: either the users never wiped the memory contents before resale or their attempts to remove data were incomplete, such that fragments are still left on the drive. This happens because when a user "deletes" content off their devices many operating systems do not actually remove the data. Instead the operating system removes the method that allows users to access the address location of that data. While, the user can no longer access the "deleted" data, their data remains written to disk until it's overwritten by new information.

The advantage of this dataset was that it was large and included real user data. While it helped us make some important inferences, it was not sufficient for our purposes because it lacked ground truth. We could not question the drive owners to see if the actual results matched our inferences. However, the Real Data Corpus data set helped to eliminate errors in our graph construction process and provide a large sample of email addresses that provided no information about the drive owner's social network. Some examples include clusters, which contained email addresses from developers of software and others which contained email addresses from different web pages.

Volunteer Drives

After getting IRB approval, we collected the Volunteer Drives from student and faculty volunteers recruited at the Naval Postgraduate School. Volunteers agreed to provide their personal devices to be imaged and analyzed in connection with this research. We were fortunate enough to have seven volunteers who provided ten drives. Following the testing of the images from their personal hard drives, they were interviewed. The interviews were conducted to learn more about the drives and the results of our analysis. The volunteers are in the best position to explain who certain email addresses belong to and if there is a connection between clusters of email addresses because the volunteers know what data they have intentionally stored on their personal drives, who they are in contact with, who uses their devices, and what applications they use. In addition, if further research was needed and the drive owner provided permission we investigated the drive sectors on the original media that contained the email addresses of which the graphs we did not understand were composed.

All of the images were from machines running proprietary operating systems. The operating systems were various versions of Microsoft Windows or Apple's OSX. In order to make copies of these systems we used two imaging tools: for machines running OSX we used the dd command line utility and for machines running Windows, we used FTK Imager.

4.1.2 Extracting Email Addresses

After the images were obtained we used Bulk Extractor to extract the email addresses. We chose Bulk Extractor because it reads the hard drive in a single linear pass ignoring the file system. We preferred this method of extracting data for two reasons. First, it is file-system agnostic. Second, by bypassing the file system, Bulk Extractor can look at storage locations that the file systems ignore, such as deleted data.

As previously discussed in Section 2.4.2, when users delete information from their devices, some users assume the data is wiped away, and that is not always true. Instead most operating systems remove a pointer to that storage location and wait for new data to overwrite the old data. If the old data is never overwritten we can extract it using forensic tools. This behavior is a benefit to us, since it allowed us to extract email addresses from these hidden locations. Another nice feature of Bulk Extractor is its ability to decompress compressed data and to produce the email addresses from these compressed files. With other tools, artifacts stored inside the compressed data would be overlooked, which might cause a failure to match the email address patterns we are looking for.

We are only interested in two of the output files produced by Bulk Extractor. The first is the email.txt file, which contains the extracted byte offset on the storage device, and a few bits of content stored before and after the email address. The second file we are interested in, is the email_histogram.txt file, which contains a histogram of all the email addresses and their frequencies.

Bulk Extractor also has the ability to accept a stop list. We used the stop_content.txt stop list available with the Bulk Extractor release at Digital Corpora [24]. This stop list was created by running Bulk Extractor on drives from the National Software Library in 2014. Using a stop list is important because many email addresses, such as the email addresses of developers who helped to create software on the device, were not relevant to the types of networks we were looking to study.

4.1.3 Constructing Graphs From Addresses and Byte Offsets

Once we had the list of email addresses from Bulk Extractor's email.txt file and their byte-offset from Bulk Extractor we were able to process the data in order to create our social network graphs. We processed the data by looking at each email address and its byte-offset on the hard drive compared to every other email address located within the byte window. We tested a distance of 128 bytes and 256 bytes.

We then created a graph by using these email addresses as nodes. If the email address was within the byte window size of the next email address, we placed an edge between the two email addresses and considered that pair of email addresses as connected. We counted the number of times that a pair of email addresses showed up on the same hard drive, and assigned edge weights equal to this count. Of the two byte window sizes, 128 bytes seemed to be more reasonable. Graphs obtained using the 256 byte window did not provide any additional useful information. On the contrary these graphs had to be further filtered to ensure that they only displayed the most relevant nodes. Otherwise there were too many nodes to quickly sort through.

We accounted for email addresses that had been compressed by making them into their own graph. These email addresses were compressed with each other because they were part of the same files and/or folders. That means they originally had a connection and it made sense that we should ensure they are part of the same graph when attempting to construct social networks.

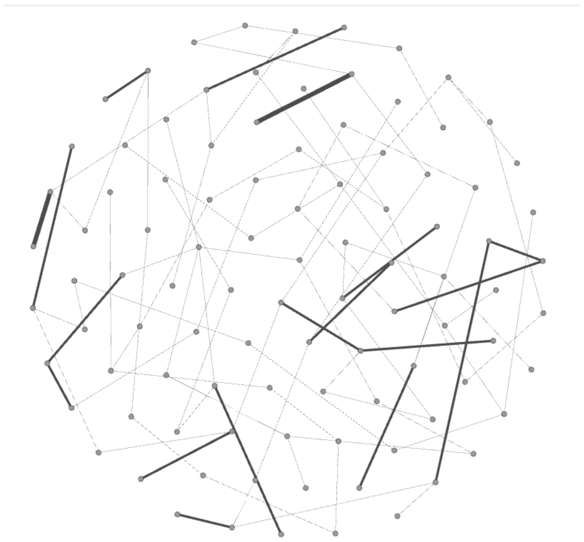
The above process of translating Bulk Extractor output to Gephi-readable graphs was automated by a program we called Betographer, which we created for this thesis project. We chose Python as the programming language because it supports a number of software packages that have been useful to our research such as NetworkX, which provides various methods of creating, storing and studying complex networks with many nodes and edges.

In addition, Betographer has the ability to break the graph of an entire hard drive into more manageable components by only returning components that have the highest number of nodes. We used this function to extract the top 20 components of each drive, which we reviewed in the exit interviews with the volunteers. Since the code is not currently available for public release, a copy of Betographer can be found in

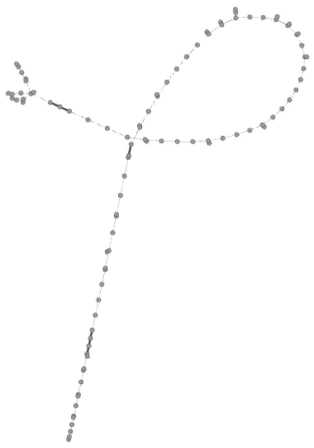
4.1.4 Visualizing the Data

Once graphs were created, we used Gephi to obtain a visual representation of the data and further manipulate the results. We realized that if the graphs did not highlight the most important information analysts would not use them. Therefore, we wanted to provide an aesthetically pleasing and useful format for analysts. Furthermore, we wanted a method that was intuitive and consistent so analyst reviewing the graphs could easily compare different graphs. It is difficult to recognize the same graph when two different visual layouts are used. In addition, we wanted our layout to minimize edge crossings so that the results did not look like a ball of yarn, and to spread out nodes so that their distance from other nodes was relative to how connected they were. So, if two email addresses have never been connected they should be placed far apart but addresses with many connections should be closer together on the graph. Finally, we wanted to spread the graph across a 2-dimensional plane so that the graphs would not appear too busy in any one area of the screen.

Based on these criteria, the layouts we believed were the most useful for our data and pleasing to the eye were Fruchterman-Reingold [25] and Force Atlas 2 [26]. These algorithms are force-directed: they work by giving the nodes opposing forces so all the nodes repulse each other. The more connections they have, i.e., the higher the weight of the edge between two nodes, the less they repulse each other. This causes nodes with high edge weights to appear near to each other on the graph. The total distance between nodes depends on the number of nodes in the graph and the total available space. The algorithm tries to spread out connected nodes as far as the given working space allows, so that nodes are not right on top of each other and can be seen clearly. The two layouts differ in the amount of empty space with which they surround the nodes, the exact distance between nodes and the values of the forces of attraction and repulsion. For comparison Figure 4.1 shows the same graph using both layouts.



(a) Realistic Data Corpus Drive with Fruchterman-Reingold Layout



(b) Realistic Date Corpus Drive with Force Atlas 2 Layout

Figure 4.1: Demonstration of Two Topologies used to Display Graphs.

4.2 Separating the Graphs into Bins

After the graphs were produced we analyzed them and separated them into different bins based on our findings. The steps we took to separate the graphs were as follows:

1. Calculate the total nodes and weights for all graphs.
2. Employ a k-core filter to identify the core of the graphs (see Section 2.1.1)
3. Visually inspect the content of the graphs for contents.
4. Sample nodes and attempt to understand them by investigating their sources on the original media.
5. Separate graphs into three different categories: Useful, Unclear, Not Useful.

After we created the graphs for each drive we began our analysis on them. We recorded the number of nodes and edges for each graph. If the graph had more than 200 nodes we filtered it to the core and analyzed the core. For the data in the Realistic Data Corpus we compared our results with the facts surrounding the various scenarios. For the data obtained from the Real Data Corpus we conducted Internet research to see if there was public information supporting the connections between the email addresses in our clusters. For the data obtained from volunteers we visually inspected the graphs, separated the graphs into bins based on what we knew about the drive owner, and later confirmed our results in the exit interview. The graphs that were clear were graphs that contained user-names that were actual names and top level domains that are typically used for personal email like “Gmail” or “Yahoo”. If we were not sure of a top level domain we conducted Internet research to see if the top level domain was an institution or agency. Graphs that were clearly not useful were those that contained developer email addresses or generic user-names such as no-reply@thisdomain.com. There were some graphs that we could not clearly label. Those we attempted to confirm with the drive owner or conduct further forensic testing.

Our "Useful" bin included graphs that looked like they showed characteristics of true social networks. According to Slawski [27] the global top ten top level domain names (TLD) are ".com", ".org", ".edu", ".gov", ".uk", ".net", ".ca", and ".de". If the graphs contained at least 75% of nodes with these TLD then we considered them likely to be part of a social network. We also reviewed the graphs to determine if the range of nodes were realistic. For instance, it is unlikely that a graph with $100,000$ nodes represents a true social network. Most people do not frequently communicate with $100,000$ other people. According to Hampton et al. [28], the average American has a Social Network of 634 contacts, those who use the Internet and cell phones having a higher number of connections respectively. The average number of Facebook friends for any one user is 338 and the median is 200 [29]. In addition, anthropologist and evolutionary psychologists Robin Dunbar, has done research on how many people can be in a person’s Social Network [30]. The numbers were a range of $100-200$ for casual friends to no more than 1500 for those people who especially adept at remembering names to match to the faces. These statistics convinced us that components or subgraphs that represent social networks should have approximately 200 nodes or less. In cases where our graphs contained more nodes, we employed techniques to attempt to isolate the components at the core.

The "Unclear" bin included graphs that we were unsure about. They included top level domains from foreign countries such as ".ru" and ".lu". While these may be common and expected on the foreign drives they were not expected to be seen in the drives from our American volunteers. The domain names and user-names were also questionable. They tended to be names that could be valid users. Examples include items such as freezer_deep@clown.ms or timothy@y7sadhj.net. For privacy issues, we are not using any of the actual email addresses found on the drives but these examples are similar in syntax to our findings. If the user did not know where the graphs came from, we conducted further analysis with the drive owner's permission. The analysis performed included Internet searches and using The Sleuth Kit to identify which file name and path the nodes came from. If the graph's origin was determined we changed its bin category. However, in many cases these results still were not clear and we had to leave them for future work.

The "Not Useful" bin included graphs that we were confident were not the type of social networks that we were interested in. These graphs did include some social networks but they were not the networks of the drive owners or their comrades. The social networks discovered in this category were groups of programmers such as developers for Microsoft, C++ libraries, Ubuntu, many other software programs and email subscriptions. An example is a graph of Microsoft developers shown in Figure 4.2. Some graphs were from groups such as Navy Times, who send out emails to their members. While this does not demonstrate their personal social network it is interesting because it identifies an organization they are associated with. Table 5.2 in our analysis chapter shows the count of graphs from the volunteer drives based on the categories: "Useful", "Unclear" and "Not Useful".

4.3 Analysis of Components Assigned to the "Useful," "Not Useful," and "Unclear" Bins

After we separated the graphs into different bins, we conducted varying levels of analysis on each bin. The following outlines the approach we took with each bin:

1. Calculate Centrality, Modularity, Weight and Degree for Graphs in the "Useful" bin.
2. Conduct Cross Drive Analysis for Graphs in the "Not Useful" bin to discover similar graphs across drives.

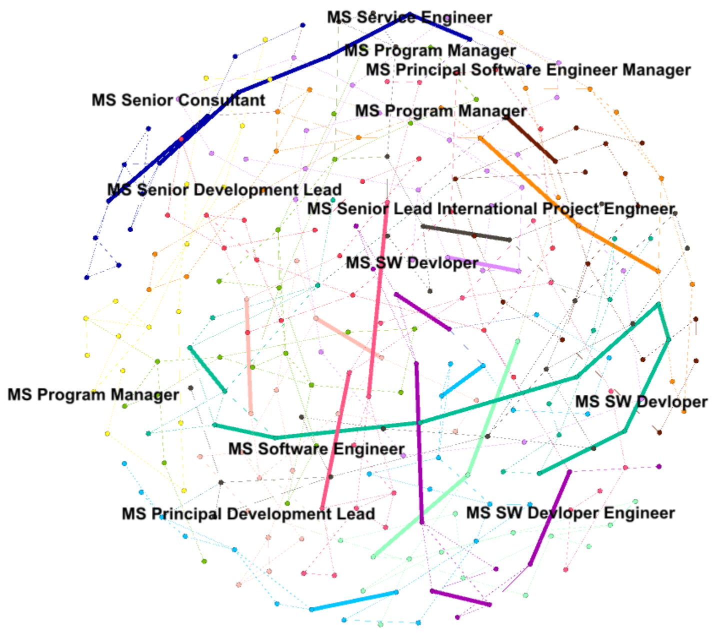


Figure 4.2: Component of Microsoft Developers: This component demonstrates a graph that was placed in the "Not Useful" bin, but could be used in future works to conduct fingerprinting. It was found on drives 2, 6, 9, and 10 as the third and/or fourth component. The components respective ID numbers are $d2c3$, $d9c3$, $d6c4$, and $d10c3$. We were able to label the developers job descriptions by conducting web searches.

For graphs that were useful we filtered the graph to the k-core, which always happened to have less than 200 nodes when filtered (except in the case of a component from a drive belonging to a web-server administrator) and began calculating statistics. We calculated the closeness centrality, betweenness centrality, eigenvector centrality, modularity, degree and weight for each node. We also calculated the average degree, average weighted degree, network diameter, graph density, modularity, and average clustering co-efficient for each graph. For graphs that were unclear we only conducted this additional analysis if, during the exit-interview, drive owners confirmed that these graphs were useful and explained who the email addresses belonged to. For graphs that were not useful we conducted visual inspections to determine if similar graphs were present on other drives, and tagged these graphs to be used in future works such as the graph in Figure 4.2. Our criteria for determining similarity was based on having a majority of nodes with the same email addresses across storage devices. A limitation of our approach is that we could not be completely certain that there were no good addresses in graphs labeled as "Not Useful" and we did not develop criteria for identifying not useful graphs automatically. Though the creation of a classification algorithm is beyond the scope of the current work, we labeled these datasets for use by future researchers in developing a systematic approach to distinguish between Useful and Not Useful graphs.

4.4 Obtaining Ground Truth with Exit Interviews

The exit interview consisted of verifying whether we were accurate in our determination of which graphs represented social networks. To answer this question, we compared the information from the exit interviews with the inferences we made before talking with the drive owners. The goal during these interviews was to discover whether the clusters represented real world relationships between individuals, or if not, whether there was some other type of noteworthy relationship that the component captured, such as a cluster of log-in addresses for access to a particular application. In addition, we wanted to gain insight into the mechanism by which errors were introduced in the graphs for example, nodes that are incorrectly identified as email addresses due to the inclusion of an "@" symbol, or find email addresses that were not important and could be disregarded; for example, email addresses from operating system developers were not considered interesting for purposes of our research, but their email addresses are valid and they form a social network as well.

We took the following approach:

1. Create PowerPoint presentation and data tables displaying results.
2. Question drive owners about storage device.
3. Verify results with drive owners.
4. Request permission for further analysis of a drive if some results were unclear.

We first created a Powerpoint Presentation and a data table for all the "Useful" components. The PowerPoint Presentations contained all 20 components from the drive owner's storage media and the data collected in Tables 5.1 and 5.4. The data tables for "Useful" components contained metrics on degree, weighted degree, eccentricity, closeness, betweenness, modularity class, and eigencentrality for each node in components that were filtered to the core if they contained more than 200 nodes. We sat down with the drive owners and showed them the graphs that we tagged as useful and confirmed whether we were accurate or not. We asked whether they had email clients, since this factor could influence the number of email addresses stored on their hard drives. We also showed them the nodes from "Useful" graphs ranked by betweenness and eigenvector scores to determine which accurately reflected the rankings of associates. We then showed them a sample of the nodes from graphs that we tagged as "Unclear" and "Not Useful." If they recognized the samples we continued our line of questioning to determine what applications were causing graphs that were not social networks. If the "Unclear" graphs were determined to be "Useful" we changed the tag and proceeded with the same line of analysis and questioning as graphs originally tagged as "Useful." Finally, we encouraged drive owners to provide any additional information or suggestions that they believed would be useful to our research or could clarify our results. A full list of our interview questions is available in Appendix C.1. For the components that were still in the "Unclear" bin we asked permission from the drive owner to conduct further forensic testing in an attempt to obtain additional information about the components.

After the comparison, we recorded three statistics: false positives (graphs that we had labeled "Useful", which interviews revealed to be "Not Useful", false negatives (graphs we labeled as "Not Useful" which interviews revealed to be "Useful", and true positives. Due to constraints on interviewer time, we did not perform an exhaustive review of the "Not Useful" graphs, especially in cases where the origin of the email addresses in the graph was easily determined. We therefore omit a true negative category, since we do not have complete verification of all graphs in this bin. Once the true relationships of the clusters were established, we recorded how many of these inferences were correct and how many were not. This was important because analysts would not always be able to talk to the drive owners to verify their results. So it was important that we could infer valid relationships reading the graphs alone. Trends that signaled true negatives were documented for future work on automation techniques to remove noise from the results. The results of this analysis are discussed in Section 5.5.

There were many instances where the relationships were unclear and we had to investigate further. The owners of the drives often could not explain why clusters of email addresses were found on their drives. Some email addresses were not recognized by the owners at all. We used The Sleuth Kit to investigate the surrounding circumstances of unknown email addresses and establish evidence to support why they were on the drive.

4.5 Further Testing Conducted

Results in the "Unclear" category tended to receive this label because we could not determine which application produced the email addresses or whether the email addresses were actual email addresses or just strings that displayed similar properties to email addresses. As previously mentioned, we attempted to question the drive owners about these graphs but in the cases where the drive owners could not provide definitive answers, we took the following steps:

1. Used The Sleuth Kit to recover files.
2. Inspect hex dumps to read context surrounding nodes of interest.

We used The Sleuth Kit to recover the files that were associated with components of interest. Once we recovered the files we used the file name and path to determine what applications were associated with "Unclear" components. If we discovered the file name and still had questions we performed a hex dump of the file and attempted to discover whether the file contents provided any helpful clues. Graphs where this process did not provide additional insight were left in the "Unclear" bin.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5: Results and Analysis

In this chapter, we discuss the results of our experiments and the analysis we conducted. We begin with the Real and Realistic Data Corpus, proceed to the Volunteer Drive dataset. After discussing the experiments and analysis, we provide a synopsis of the most interesting graphs. Finally, we examine the effectiveness of well-known centrality measures at predicting the relative importance of members in our social networks.

5.1 Results From the Real Data Corpus and the Realistic Data Corpus

Our preliminary analysis of the Realistic Data Corpus and Real Data Corpus drives revealed some general characteristics of the drives, such as what operating systems or programs were installed. An example is shown in Figure 5.1, demonstrating a component that indicates Ubuntu was installed on the storage media. Because this image was from the Realistic Data Corpus [23] we were able to confirm our results. In addition, we found components that "looked like" valid social networks based on the top level domains in the addresses, and whether the user-names were valid names or generic handles such as "no-reply". However, for the Real Data Corpus we could not determine whether the networks were created from actual associates of the drive owner, since we did not have actual drive users to verify the information. Note that, in the interest of protecting the privacy of the original device owners, we do not include a graph that demonstrates this problem. On the Realistic Data Corpus drives we were able to confirm that some of the heavily weighted nodes matched the characters in the scenario. An example is shown in Figure 5.2, where there were four main characters in this particular scenario. The characters were CEO Pat McGoo, IT administrator Terry Johnson, and two patent researchers Jo Smith and Charlie Brown [31].

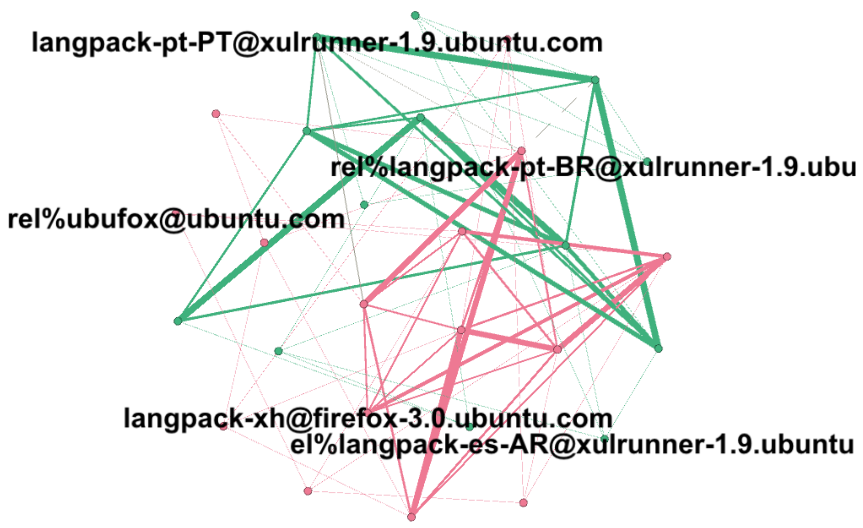

langpack-pt-PT@xulrunner-1.9.ubuntu.com
rel%langpack-pt-BR@xulrunner-1.9.ubu
rel%ubufox@ubuntu.com
langpack-xh@firefox-3.0.ubuntu.com
el%langpack-es-AR@xulrunner-1.9.ubuntu

Figure 5.1: Component from UBNIST1: UBNIST1 is a drive from the Realistic Data Corpus. Displayed is the 3rd component from this drive. UBNIST1 was created from a USB memory stick that contains a bootable copy of Ubuntu 8.10 Linux [23]. Without referring to the history of the drive we could tell Ubuntu was installed based on the domain of the nodes. The component displayed is unfiltered.

5.2 Analysis of Aggregate Statistics on Graphs from Volunteer Drives

Table 5.1 presents basic information from the drives that we analyzed. We assigned ID $d1$ through $d10$ to each graph in the data set, corresponding to the ten drives from which the graphs were created. Components of a graph are referred to by their graph ID concatenated with the letter c and a number corresponding to the component's relative size within the graph—that is, $c1$ is the largest component, $c2$ the second largest, etc. Thus $d1$ refers to the entire graph produced from the first volunteer drive, and $d1c19$ refers to the 19th largest component of that graph, if there is one. As mentioned previously the nodes are the email addresses discovered on the drives and the edges are a relationship between those nodes. Relationships are said to occur if the email pairs are discovered within 128 bytes

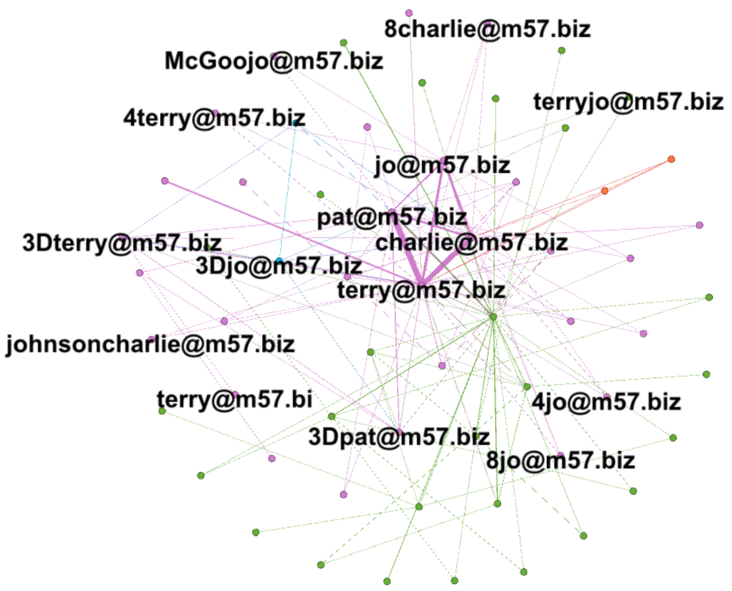


Figure 5.2: Component from terry-2009-12-11-001.E01: This image is from the m47 patents scenario. The image is part of the Realistic Data Corpus. Displayed is the 2nd component from this image. The component displayed is unfiltered. The nodes with the thickest edge weights belong to the main characters of the scenario.

The last two columns of Table 5.1 give the percentages of nodes and edges that were analyzed in relation to the total number of nodes and edges on the graphs built from the drive. While we only analyze a small percentage of the nodes, we are able to analyze the majority of the edges. For our research it was not important to analyze all the email addresses on the storage medium. Instead, we needed to examine the addresses that had links to other addresses. We are seeking to determine relationships between email addresses, and that is accomplished by extracting and analyzing a majority of the edges on the storage medium. The large number of unexamined nodes is indicative of the fact that most of these were singletons, or part of very small components.

Attempting to view a graph of an entire storage device was not useful unless the graph held less than 200 nodes. It is similar to trying to hear a choir of 30 people while 100,000 people are singing different songs at the same time. The most important

Graph ID	Operating System	Drive Capacity	# of Nodes	# of Edges	% of Nodes Studied	% of Edges Studied
d1	OSX	320G	55030	68594	30%	77.5%
d2	Windows 10	100G	25101	30472	43.86%	89.99%
d3	OSX	500G	105457	172013	25.57%	43.19%
d4	OSX	320G	105482	107203	27.1%	69.82%
d5	Windows 8	20G	885	95	8.93%	83.16%
d6	Windows 8	460G	98086	458255	31.16%	80.21%
d7	Windows 7	500G	29727	50323	45.05%	95.03%
d8	Ubuntu/BackTrack	200G	42452	43535	35.75%	88.39%
d9	Windows 7 Enterprise	400G	2623	969	23.60%	95.15%
d10	Windows 7 Enterprise	150G	52327	74952	41.28%	88.24%

Table 5.1: Drive Capacity & Extracted Information for Volunteer Drives. Note that $d5$ and $d6$ were obtained from the same laptop; $d5$ was a small SSD and $d6$ was a larger magnetic spinning disk. The user stated that the laptop had dual hard drives so that the OS could be stored on the SSD for quicker data retrieval. However, upon analysis the OS was discovered on the HDD and not the SSD. The entry in the OS for $d8$ indicates that it originally had Ubuntu installed; subsequently the drive owner re-imaged the computer and installed BackTrack as the new OS.

nodes do not stand out when there are thousands of nodes or more. Furthermore, during testing when we attempted to run a layout algorithm on networks with over $100,000$ nodes, Gephi would crash. We ran Gephi on a computer with an 8GB of RAM, which allowed us to reconfigure Gephi to use 2800MB of memory; this helped with some graphs of tens of thousands of nodes, but does not represent a scalable solution. In addition, Gephi documentation states that the program was built for graphs up to $100,000$ nodes [32], which explains why graphs above this number tended to crash.

This is why we did not attempt to visualize a graph of the entire drive and instead broke them down into 20 components each, as described in Section 4.1.3. These graphs were chosen by the highest number of nodes. Because all nodes in a component must be connected by at least one path, membership in the same component implies a relationship. In Table B.1 we show the node and edge count of the top 20 graphs of each drive, with their node and edge count.

The top component of each drive contained an average of $12,774$ nodes with a range between 10 and $28,658$. For the second largest component, that number was drastically reduced to an average of $2,678$ nodes with a range between 8 and $10,192$ nodes.

Both the average number of nodes and the range fell off rapidly: for the 20th component the average was 23 nodes and the range was between 2 and 38. In a similar trend, the top component for each drive averaged $64,089$ edges with a range between 24 and $365,054$ edges, but this number dropped to an average of $9,045$ edges with a range between 7 and $39,406$ for the 2nd component and an average of 41 edges with a range between 1 and 66 edges for the 20th component. These results indicate that the majority of email addresses and relationships between email address pairs will be discovered within the top 20 components.

Graph ID	Useful	Unclear	Not Useful
d1	1	5	14
d2	1	4	15
d3	2	13	5
d4	3	14	3
d5	8	0	12
d6	0	9	11
d7	2	0	18
d8	1	0	19
d9	3	0	17
d10	1	0	19

Table 5.2: Number of Social Networks Discovered on Graphs: The "Useful" bin contains components that displayed social networks, the "Unclear" bin contains components that drive owners were unsure of and our Internet searches and forensic testing did not produce clear results, the "Not Useful" bin contains components that did not display social networks. The "Not Useful" bin contained items that included applications and developer email addresses. It was interesting that $d5$ contained many useful components while $d6$ did not contain any. Drive 5 was a SSD and drive 6 was a HDD.

We wanted to determine what statistical scores assigned to the graphs best reflected social network groups. Table 5.3 is a table of the statistics taken for each drive before any filtering was applied. We had twenty graphs per drive but most graphs did not represent social networks. Table 5.2 shows the number of components that fell into each of our bins across all drives. With the exception of $d6$, all the drives contained some social networks within the first 3 components. As indicated previously, components are sorted by node count, from largest to smallest. Table 5.3 shows all graphs that did represent actual social networks as confirmed by the drive owner. In our informal preliminary analysis, we detected no single metric that correlated clearly with social networks. There were many usernames and domains that appeared to be valid social networks but having above 200 nodes added to the noise, which made it difficult to quickly determine those email addresses that were closely associated. We did not collect statistics on "Not Useful" components. A more systematic classification scheme across a larger dataset may be more revealing; however, this first requires our methodology to be extended to a larger number of volunteers.

Note that metrics for components extracted from $d5$ are somewhat atypical, in that they have scores of 0 for several of the columns. The primary reason is due to the small number of email addresses discovered in each component of $d5$. As mentioned, $d5$ was a small capacity SSD which is believed to hold caches of user data based on the contents of its components and forensic testing. We attempted to actually find these email addresses by using the OS but they were inaccessible through the file system and upon conducting forensic testing we discovered they were located in unallocated space.

Component ID	Nodes	Core	Average Degree	Average Weighted Degree	Diameter	Density	Modularity	Average Clustering Coefficient	Average Path Length
d1c2	6087	19	8.82	29.19	51	0.10%	75.30%	44.40%	5.23
d2c2	867	8	4.08	53.40	22	0.50%	43.00%	29.90%	4.40
d3c1	24075	22	5.37	29.36	196	0.00%	62.40%	58.90%	10.51
d3c17	39	3	3.13	3.13	24	8.20%	69.90%	54.30%	8.94
d4c2	10192	6	7.70	30.00	134	0.10%	77.50%	48.20%	6.85
d4c4	328	14	7.70	23.00	8	2.40%	72.80%	47.10%	3.66
d4c7	123	2	2.72	5.43	37	2.20%	84.20%	57.70%	12.87
d5c1	10	4	4.80	8.00	4	53.30%	17.20%	72.20%	1.69
d5c5	6	1	4.67	3.00	2	33.30%	0.00%	0.00%	1.67
d5c6	5	1	1.60	6.80	2	0.40%	0.00%	0.00%	1.60
d5c7	4	1	1.50	1.50	2	0.50%	0.00%	0.00%	1.50
d5c9	3	1	1.33	2.00	2	0.67%	0.00%	0.00%	1.33
d5c14	3	2	2.00	4.00	1	100%	0.00%	100%	1.00
d5c17	2	1	1.00	4.00	1	100%	0.00%	N/A	1.00
d5c20	2	1	1.00	4.00	1	100%	0.00%	N/A	1.00
d7c1	534	5	3.07	8.98	16	0.60%	59.80%	76.90%	3.25
d7c2	9	3	4.67	4.67	3	58.30%	20.40%	76.30%	1.50
d8c2	1630	5	2.81	18.82	96	0.20%	49.50%	30.60%	8.16
d9c1	7655	3	2.40	14.07	12	8.30%	59.90%	14.10%	5.01
d9c4	233	4	5.64	11.40	65	2.40%	83.60%	60.50%	23.78
d9c11	30	3	2.55	12.36	11	12.10%	47.40%	22.40%	4.60
d10c1	19953	21	5.33	42.48	69	0.20%	95.00%	59.20%	24.72

Table 5.3: Metrics Collected from Graphs that were Social Networks

Even after separating graphs into components, some graphs were still in excess of 200 nodes. To reduce this number to a more manageable level we filtered the graphs to their core, after which the social network graphs were no more than 200 nodes, with the exception only of $d9c1$. These statistics are displayed in Table 5.4. This component $d9c1$ still has more than 200 nodes but it displays more than one social network that needs to be separated. The drive owner of Component $d9c1$ was an administrator for an email server, and would receive emails and error messages for

Component ID	Nodes	Edges	Average Degree	Average Weighted Degree	Diameter	Density	Modularity	Average Clustering Coefficient	Average Path Length
d1c2	50	650	26.00	166.52	2	0.53%	.496%	.59%	1.47
d2c2	10	43	8.60	23.00	2	0.96%	.08%	.95%	1.04
d3c1	37	532	28.76	2189.78	2	.80%	.05%	.83%	1.20
d3c17	39	3	3.13	3.13	24	8.20%	69.90 %	54.30%	8.94
d4c2	20	159	15.9	254.30	2	0.001%	.21%	.84%	1.16
d4c4	77	267	6.94	19.80	6	.091%	.643%	.44%	3.03
d4c7	123	2	2.72	5.43	37	2.20%	84.20%	57.70%	12.87
d5c1	10	4	4.80	8.00	4	53.30%	17.20%	72.20%	1.69
d5c5	6	1	4.67	3.00	2	33.30%	0.00%	0.00%	1.67
d5c6	5	1	1.60	6.80	2	0.40%	0.00%	0.00%	1.60
d5c7	4	1	1.50	1.50	2	0.50%	0.00%	0.00%	1.50
d5c9	3	1	1.33	2.00	2	0.67%	0.00%	0.00%	1.33
d5c14	3	2	2.00	4.00	1	100%	0.00%	100%	1.00
d5c17	2	1	1.00	4.00	1	100%	0.00%	N/A	1.00
d5c20	2	1	1.00	4.00	1	100%	0.00%	N/A	1.00
d7c1	6	15	5	14.33	1	1%	0%	1%	1
d7c2	9	3	4.67	4.67	3	58.30%	20.40%	76.30%	1.50
d8c2	56	205	7.321	87.429	7	.13%	.484%	.70%	3.49
d9c1	1240	13353	21.537	412.839	5	.02%	.82%	39%	2.96
d9c4	54	149	5.52	11.19	10	.10%	.69%	.65%	3.57
d9c11	30	3	2.55	12.36	11	12.10%	47.40%	22.40%	4.60
d10c1	33	447	27.09	3215.21	2	.85%	.25%	.85%	1.15

Table 5.4: Metrics Collected from Social Networks Discovered on Disk Images after Filtering to the Graph Core, for components containing more than 200 nodes.

all clients on the server. Attempting to separate graphs similar to these into email clusters and communities could be a subject of future research. Metrics for all cores are displayed in Table 5.4.

For most of the graphs, filtering to the core and recording edge weights provided enough information to get a sense of the drive owners close associates (non-extended family members, friends, coworkers in the same department) but for others it was beneficial to decrease the k value to a few degrees below the core. For example, in Component $d2c2$ the core showed members of the drive owner's soccer team. However, for that drive if the k-value was reduced by one members of the drive owner's family started to emerge. The same can be said for Component $d4c2$. At the core this graph displayed business associates; once the k-core was reduced family members started to appear.

5.3 Comparison of Researcher Inferences with Ground Truth for Graphs Determined from Volunteer Drives

We obtained encouraging results with our new method for analyzing relationships between email addresses, despite being cautious when classifying components as “Not Useful.” While the precision was only .5 our recall was .955. Meaning, of components that we initially labeled social networks, only 50% were confirmed during the interviews. However, less than 1% of true social networks discovered were labeled incorrectly. Our negative predictive value was much higher at 0.991. That is, of the components we initially labeled as not being social networks, 99% were confirmed by the interviews. We calculated these scores ignoring components classified as “Unclear”, because we were unable to determine whether components classified as “Unclear” represented valid email addresses. Table 5.5 shows a confusion matrix of our results. The one component that we incorrectly classified as not being a social network was $d9c11$. All domain names in Component $d9c11$ (except the drive owner’s email address) belonged to one particular business. We incorrectly labeled it as "Not Useful" for the same characteristics that made it actually useful. Our initial assumption was that components overly dominated with organizational names came from applications or web-services.

		Ground Truth After Exit-Interviews		
		Useful	Not Useful	Total
Researcher Inferences	Useful	21	21	$21 + 21$
	Not Useful	1	112	$1 + 112$
	Total	$21 + 1$	$21 + 112$	$N = 155$

Table 5.5: Confusion Matrix displaying actual and predicted values for components that demonstrated social networks and those that did not. Note, that components that were unclear were excluded from this table.

5.4 Structural Analysis of Selected Graphs from Volunteer Drives

In this section we will highlight some of the most interesting graphs we encountered. We start with two graphs, $d3c1$ and $d8c2$, that demonstrate classic properties of social networks. We move on to discuss $d5c1$, a component comprised exclusively of usernames belonging to the drive owner and the drive owner's close associates. Next we examine $d2c2$, $d4c2$, and $d1c2$, three graphs that show internal communities, followed by $d9c1$, which shows many networks within the same component. Finally, we show a graph, $d4c4$, constructed from identifiers that were extracted directly from the local cache of a social networking application.

We begin with component $d3c1$, displayed in Figure 5.3. This component shows university professors who were employees of the same department. The drive owner used two email clients: Mutt and Outlook for Mac. The drive owner stated that the mail clients were only used to send university related emails. The drive owner's spouse used the computer to send and receive web based mail. None of the spouses email addresses were stored locally, however, and they were not discovered in any of our graphs. The drive owner suggested that a few of the email addresses found were from professors that had not been emailed in years. Since email clients store data persistently and locally, the storage media could potentially have this information for years, while web based email stores data remotely and usually writes data to a local cache, which is quickly overwritten. This evidence leads us to believe that these email addresses were made available due to the drive owners use of email clients.

This component demonstrates several properties that are characteristic of social networks. First, the core of the graph corresponds closely with the drive owner's closest associates, and collaborators. Although the whole component includes many co-workers in the same organization, the core contains only the owner's close collaborators and members of his home department. Second, the owner was easily identifiable visually because of high degree and high weight edges connecting to other important members of the department. Finally, measures such as Eigenvector Centrality and Betweenness Centrality create a ranking that accurately places the drive owner and his supervisors at the top of the list. We discuss the effectiveness of these measures further in Section 5.5.

Component $d8c2$ shared many of the same properties as $d3c1$. The drive owner was president of a home owners associate board. The graph shown in Figure 5.4, produced from component $d8c2$, displays relationships between neighbors who were members of

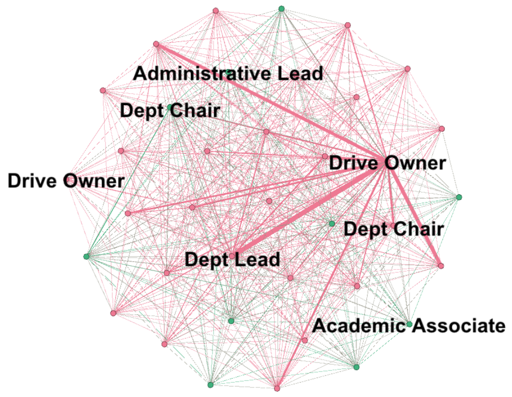


Figure 5.3: Component $d3c1$: Displays a network of Professors employed at the same University. The graph is colored by degree. The modularity class was the same for every node in this component. The modularity of the graph was low at $.049$ and the edge weights were extremely high compared to other graphs studied. The average edge weights were 2189.78. Further statistics for this graph are given in tables 5.4 and 5.3

the board. The drive owner frequently received emails from his neighbors. The edges with heavy weights connect nodes representing the drive owner, the owner's spouse and a neighbor who was also a co-worker and succeeded the drive owner as board president. As in Figure 5.3, the most important figures in the network are rapidly identifiable by visual inspection based on edge weight and degree.

The presence of multiple email addresses representing the same person is a common occurrence, and in some components this feature dominates the entire graph. Component $d5c1$, shown in Figure 5.5, shows an example of this phenomenon. The graph contains only email addresses frequently used as usernames to log into various online applications. The drive owner stated that while most of the email addresses did belong to the drive owner, others included the owner's mother, mother-in-law, and spouse. The drive owner states that they did not use any of their email addresses to send email messages to their mother or mother-in-law. These email addresses were mostly likely stored when they logged into online applications that required a username. Many of the nodes are the same exact email address with a small prefix concatenated to the front of the email address such as "email", "eml_address" and "emailContactAddress". This component was interesting because it demonstrated that work could be done

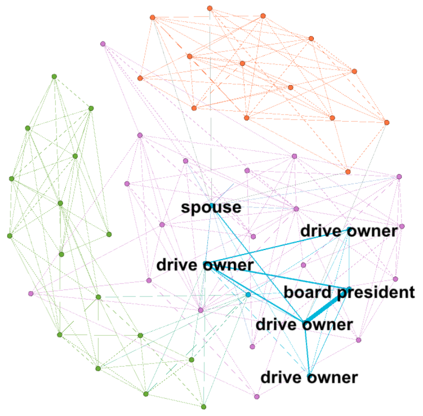


Figure 5.4: Component $d8c2$: Displays a network of members of a neighborhood board for which the drive owner had previously served as president. The core (at $k = 5$) displayed 56 nodes with 205 edges. Heavy edge weights connected the drive owner and spouse, as well as the drive owner and the board president who succeeded the owner. The multiple nodes labeled drive owner are a result of multiple email addresses. The nodes are colored by modularity class. Further statistics for this graph are given in tables 5.4 and 5.3

to discover email addresses that individuals use as usernames to log onto different applications.

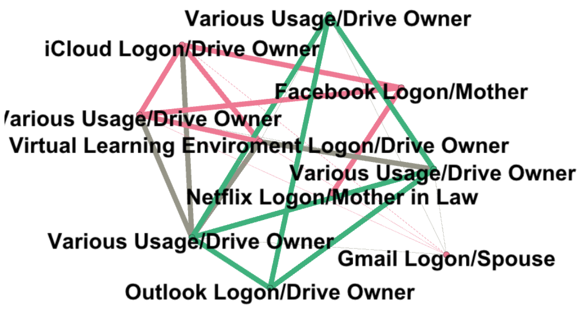

Various Usage/Drive Owner
iCloud Logon/Drive Owner
Facebook Logon/Mother
'arious Usage/Drive Owner
Virtual Learning Enviroment Logon/Drive Owner
Various Usage/Drive Owner
Netflix Logon/Mother in Law
Various Usage/Drive Owner
Gmail Logon/Spouse
Outlook Logon/Drive Owner

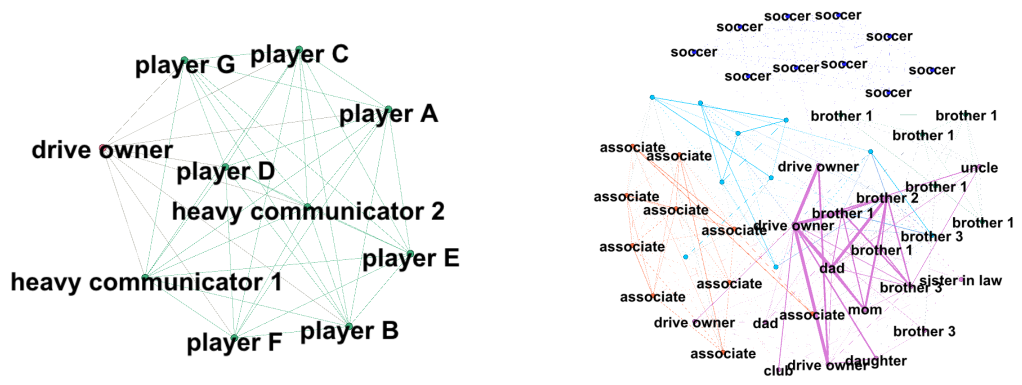
Figure 5.5: Component $d5c1$: Displays a network of email addresses used to log into online application by using these email addresses as the username. Further statistics for this graph are given in tables 5.4 and 5.3

Figures 5.6, 5.7, and 5.8 differ from graphs presented so far in that they display components that can be broken up easily into smaller communities. In addition, these components were more interesting when k-core value was reduced to include more nodes that just the core. Both of these characteristics can be seen in component $d2c2$, displayed in Figure 5.6. This drive owner used Outlook and Yahoo to send and receive mail. The thick edges were connect nodes representing the drive owners, the coach, and a teammate who responded to team messages more than others. The drive owner explained that he did not frequently send emails to the team but was a recipient of these group messages. This graph is notable in that it provides an example of a situation where a noisy sub-community can create a distorted image of the owner's social network by overwhelming other communities. For most of the graphs we inspected, nodes outside the core they tended to be email addresses from people or agencies that the drive owner was in contact with less frequently. However, in this graph, although the core occurred at $k = 8$, the 7-Core also included useful information—in particular, nodes that represented friends and family of the drive owner. Even though the drive owner's family was outside the core, they still appeared at the top of the ranking produced according to the centrality measures we ran, when k-value was reduced. Again, see Section 5.5 for further discussion of the effectiveness of centrality measures.

Nodes	Modularity Class	Degree	Weighted Degree	Closeness	Betweenness	Eigenvector
Drive Owner	0	8	22	.9	.125	.906
Heavy Communicator 1	1	9	29	1.0	.25	1.0
Heavy Communicator 2	1	9	21	1.0	.25	1.0
Player A	1	9	29	1.0	.25	1.0
Player B	1	9	27	1.0	.25	1.0
Player C	1	9	25	1.0	.25	1.0
Player D	1	9	25	1.0	.25	1.0
Player E	1	8	19	.9	.125	.91
Player F	1	8	17	.9	.125	.91
Player G	1	8	16	.9	.125	.91

Table 5.6: Statistics for Individual Nodes of a Soccer Team, Filtered to the Core

Component $d4c2$, shown in Figure 5.7 shows a group of friends, family, coworkers, and classmates. As with component $d2c2$, we lost vital information when we restricted our examination to the core of the network (at $k = 14$). We illustrate this by showing the core and the 10-core side by side in Figure 5.7. The graph at its core displayed a network of the owner's classmates, and when dropped to k-core 12-13 additional

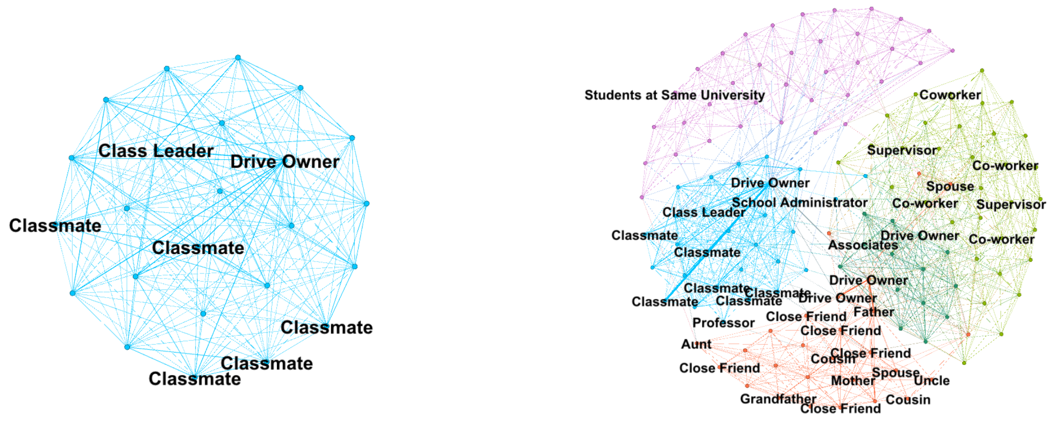


(a) 8-Core of Component $d2c2$ (b) 7-Core of Component $d2c2$

Figure 5.6: Component $d2c2$: Displays a network of soccer teammates. Once filtered to the core at k=8, it displayed 10 nodes—all members of the same soccer team. The teammates who emailed the group most frequently are labeled as heavy communicators. The nodes are colored by modularity class. Below in table 5.6 are some calculations used to compare nodes. When the core was reduced to 7 the graph increased to 51 nodes and the drive owner's family members started to emerge. Further statistics for this graph are given in tables 5.4 and 5.3

classmates were added to the graph; these are all colored in light blue. At k-core 11 the drive owners co-workers and associates were added to the graph, colored light and dark green respectively. The associates were people the drive owner knew but did not consider close friends. At k-core 10 close family members and friends were added; they are displayed in orange. At this core, displayed in pink, are other students who attend the same university as the drive owner. However, the drive owner did not know these students personally and believed they were on the hard-drive due to being part of the same distribution lists from school administrators. Similarly to $d2c2$, even though the drive owner's family was outside the core, they showed up at the top of Centrality measures, when k-value was reduced. An example of the Centrality measures for this drive is in Table 5.7.

Below a k-core of 10, a large number of email addresses were introduced that the drive owner did not recognize. As in components previously discussed the heavy edges did in fact represent people the user communicated with most often. Also, here again



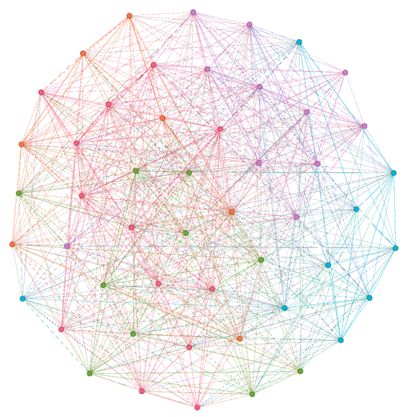
(a) The network at the core

(b) The network with $k = 10$

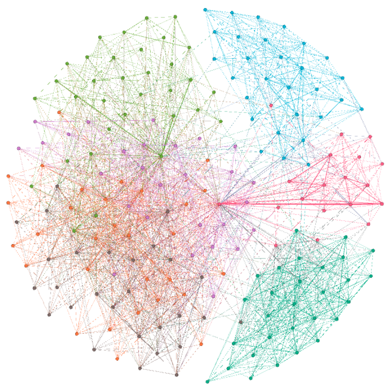
Figure 5.7: Component $d4c2$: Displays the drive owner's associates. In figure (a) the graph is filtered to the core, which was 14. At the core this component only shows a network of classmates. Once reduced to k-core 10, shown in figure (b) the component shows other associates of the drive owner. The modularities we calculated for this graph at the core, at k-core 10, and unfiltered were .247, .641, and .771 respectively, suggesting that this component could be partitioned into communities. Further statistics for this graph are given in tables 5.4 and 5.3

the graphs show multiple nodes for the drive owner. There are many nodes labeled as the drive owner's, each one represented a different email addresses that the drive owner had. Notably, in this case the different email aliases are situated in different communities, suggesting that the owner used different addresses to communicate with different groups.

In component $d1c2$, we again see distinct communities emerge outside the graph's core. The core at $k = 19$ and the 10-core are displayed in Figure 5.8. This component was extracted from the drive of a university student who was a part of many organizations. The student used Outlook to send and receive mail. The student confirmed that all the graph nodes were members of the various university organizations. However, these members were associated with the student via group emails and never personally introduced. In Figure 5.8, we show component $d1c2$ with k-core reduced to 10 because this graph also demonstrates a potential for community detection.



(a) The network at the core



(b) The network with $k = 10$

Figure 5.8: Component $d1c2$: Displays a loose network of students who were members of various University clubs that fell under the same higher organization. The component displayed is filtered to the core, which is 19. It had nodes, edges, and a modularity of 50, 650, $.496$ respectively. The nodes are colored by degree. The nodes that are not colored in red all belong to the drive owner. Further statistics for this graph is given in tables 5.4 and 5.3

The next series of graphs comes from $d9$ and we will talk about Components $d9c1$, $d9c4$, and $d9c11$ from this drive. This storage device belonged to a user who was also the administrator for a web-server running on the device. Therefore, the owner had email addresses and connections that belonged to personnel who did not access his machine directly. The server was set up to send copies of emails to the administrator's in-box, when that specific email account had turned on its "away messages." The business was using Dovecot to run their email server and SquirrelMail to access their in-boxes. Dovecot is an open source POP3 email server for Linux systems. The business also conducted web-hosting services for one of their clients. Component $d9c1$ shown in Figure 5.9 is a graph from a business owner's drive; it shows employees and clients.

Component $d9c1$ had too many nodes to be visually useful without further filtering, but still displayed a number of striking characteristics. The drive owner confirmed that this graph had email addresses from different clients and employees of the organization. This graph would be a good candidate to break up into clusters of smaller

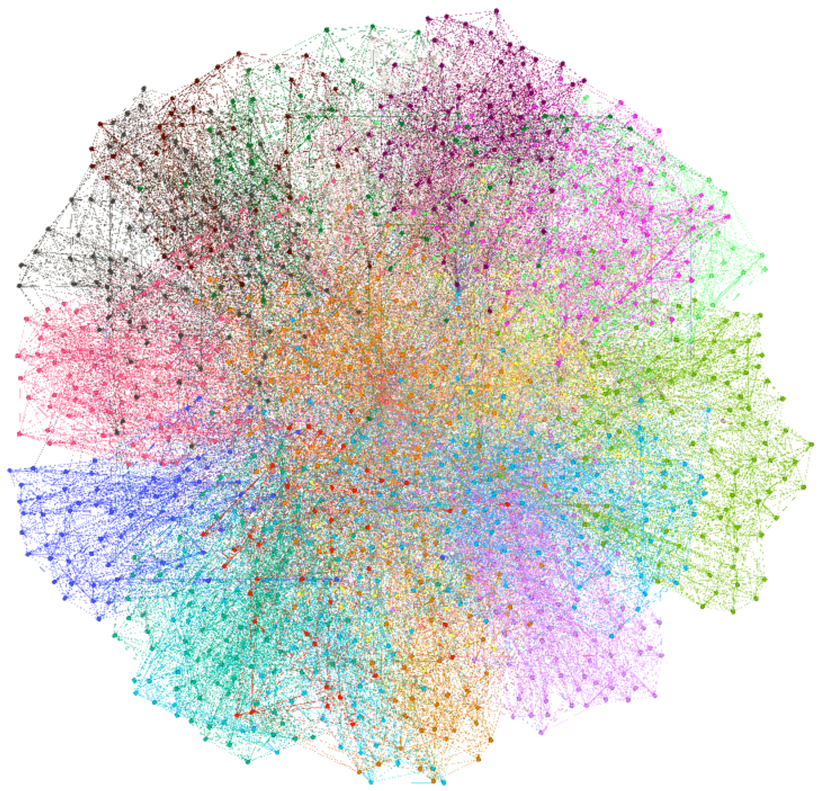


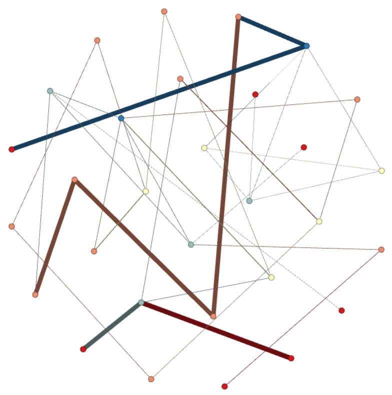
Figure 5.9: Component $d9c1$: Displays a network of employees and clients. This figure shows the component filtered to its core of 14. Notably, it's core has a far more complex structure than the cores of most other graphs we analyzed. This component is a good candidate for future research in community detection. The graph had a modularity of $.87$ with no filter and a modularity of $.821$ at the core, both of which scores are unusually high and indicate that the communities are very distinct. The nodes in this figure are colored by modularity class to show this strong partitioning. Further statistics for this component are given in tables 5.4 and 5.3

.

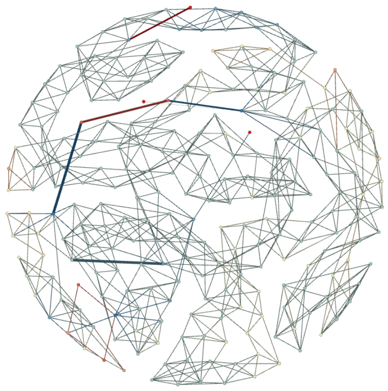
communities, as discussed in Future Work (Section 6.2). The drive owner stated there were 20 clients, which matches closely to the 19 modularity classes identified. In addition, this graph had a modularity of .821 at the core suggesting that groups did not cross communicate. As a result of conducting web-hosting services, components $d9c11$ and $d9c4$ contain information about the social network of the web-hosting client. Two graphs belonging exclusively to the client's organization are demonstrated in Figure 5.10. These graphs were included because it was interesting to find social networks that belonged to groups that did not include the drive owner. However, because these email addresses did not belong to anyone in the drive owner's personal social network there was not a lot of additional information we could obtain about these components. The drive owner was only able to offer that these were employees of a particular company.

We note, however, that these two graphs show a visually distinct structure that differs from that of the social networks shown in previous figures. Although there are many edges, few nodes have very high degree—most are connected to only a few neighbors in a long chain of cliques. For example, in $d9c4$ the highest degree node is 8, and the vast majority have a degree of 5, indicating that they communicate only with a few neighbors. In some cases, these connections are repeated many times, which is what produces the heavy-weighted paths running across the graph. In most other cases, the edge weights are uniform. We inspected the underlying disk images and found that these graphs were both produced from zipped files that Bulk Extractor had unzipped. Component $d9c4$ was found in unallocated space, but Component $d9c11$ was associated with a path that suggested it was a database (the letters "db" were included in the file name). We suspect these graphs may come from files that store electronic address books.

The last graph of interest is Component $d4c4$, which is displayed in figure 5.11, was a graph of Facebook user-ID's. Superficially, this component seems to share some of the visual markers of components $d9c11$ and $d9c4$; however, the further analysis reveals that the structure is more complicated—we see a larger range of degrees and connections across different IDs, which suggest repetition of IDs in different contexts on the underlying storage media. Although the user-IDs were all of "friends" on Facebook of the actual owner of the laptop, none of our centrality metrics gave an accurate picture of the friends that the owner communicated with most frequently. The drive owner stated that while they knew all the users in this graph, they rarely if ever communicated with those who had the heaviest edge weight. The addresses were extracted from unallocated space, but it is possible that they are the result of a local cache for the Facebook web application.



(a) Social Network of Client 1 (component $d9c11$)



(b) Social Network of Client 2 (component $d9c4$)

Figure 5.10: Components $d9c11$ & $d9c4$: Displays two components from $d9$, unfiltered. These contained 30 and 233 nodes respectively of email addresses belonging exclusively to client organizations. The nodes are colored by degree. Further statistics for these graphs are given in Tables 5.4 and 5.3

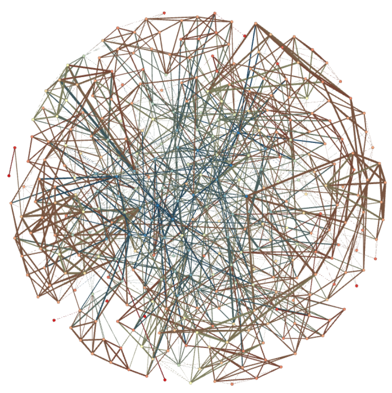


Figure 5.11: Component $d4c4$: Displays the drive owners Facebook "friends". This is a component from drive 4. While the modularity was $.643$, the drive owner could not explain why any set of people would be grouped together. The entire component as a whole made a sensible set. The entire component was comprised of nodes that held the user-IDs of the drive owners Facebook "friends". The nodes are colored by degree. Further statistics for this graph is given in tables 5.4 and 5.3

5.5 Discussion of Centrality Measures and Their Effectiveness

In addition to identifying and examining social networks, we also wanted to determine whether we could automatically discover people who were leaders of these extracted groups. Upon completion of our exit interviews with the drive owners we determined that Eigenvector centrality best reflected the most important personnel in the group. Those that the drive owner communicated with most frequently had an Eigenvector score of close to one. In cases where the components contained more than 200 nodes, we calculated the Eigenvector and Betweenness centralities for each node after filtering graphs to their cores. For graphs which contained less than 200 nodes we did not filter to the core since 200 is an acceptable number of associates to have. Though Eigenvector centrality was more accurate per exit interviews with the drive owners, these scores were very similar, often with the same top twenty associates, although the order was slightly different. As Betweenness centrality is more expensive to compute, we conclude that the Eigenvector centrality is indeed the best measure for these networks. Closeness centrality was also calculated on every node and included in our data tables, but this metric was not reviewed in detail with drive owners. Since centrality is calculated on every node we do not include these have statistics in our previous tables. A sample of the centrality calculations is given in Table 5.7 for the top 20 nodes with the highest centrality scores in drive $d4c2$. Our preliminary results demonstrate this to be a promising area for future work, in order to determine if Eigenvector centrality can indicate close associates and if the convergence of scores between Eigenvector and Betweenness indicate valid social networks.

Ranked by Eigenvector	Ranked by Betweenness	Ranked by Closeness
Drive Owner(1)	Drive Owner(2257.290)	Drive Owner(.547)
Father(.684)	Father(1527.947)	Father(.537)
Classmate A(.654)	Drive Owner Alt Email 2(863.901)	Drive Owner Alt Email 2(.515)
Classmate B(.623)	Unknown Email Address 1(581.414)	Drive Owner Alt Email 3(.476)
Drive Owner Alt Email 1(.603)	Classmate Q(467.205)	Coworker different depts(.468)
Classmate C(.603)	Classmate K(378.889)	Classmate A(.465)
Classmate D(.599)	Department Store(332.431)	Drive Owner Alt Email 1(.456)
Classmate E(.599)	Classmate A(310.950)	Unknown Email Address 2(.453)
Classmate F(.595)	Unknown Email Address 2(277.630)	Schoolmate not in same class(.450)
Classmate G(.593)	Classmate D(272.832)	Classmate J(.450)
Classmate H(.573)	Aunt(272.697).	Classmate Q(.447)
Classmate I(.572)	Drive Owner Alt Email 3(248.521)	Classmate B(.444)
Classmate J(.553)	Classmate L(229.499)	Classmate P(.441)
Classmate K(.546)	Cousin(228.484)	Classmate D(.438)
Classmate L(.536)	Classmate P(223.538)	Unknown Email Address 1(.437)
Drive Owner Alt Email 2(.532)	Coworker same dept 1(220.764)	Department Store(.434)
Classmate M(.530)	Classmate B(215.833)	Spouse(.432)
Classmate N(.525)	Classmate M(210.764)	Classmate E(.430)
Classmate O(.522)	Classmate E(195.212)	Coworker same dept(.430)
Classmate P(.521)	Unknown Email Address 3(190.734)	Unknown Email Address 3(.430)

Table 5.7: This table shows the variations between most important node ranking by centrality metrics.

CHAPTER 6: Conclusions and Future Work

In this final chapter we present an overview of the goals, results, and contributions of our research. We also answer the questions asked in chapter 1. In addition, we present some suggestions for continued research in this area.

6.1 Conclusions

We wanted to assist analysts who depend on the ability to understand the relationships between people of interest to their investigations. The goal of our research was to present a new method of analyzing correlations between email addresses, in order to make analysts more efficient during investigations and decrease their workload. We accomplished this goal by automatically extracting email addresses and creating graphs depicting the relationships between them, then showing that these relationships correspond to real-life social network information.

Our work extended beyond simply discovering social networks. In addition to the social networks of many drive owners, we also observed installed OS files and applications, and usernames used to log onto specific web applications. In components where social networks were confirmed, we saw evidence of specific communities within social networks that could be detected and isolated. We identified four interesting types of graphs:

1. Components that displayed classic properties of social networks, such as high centrality scores at the core and high edge weights and degree for email addresses belonging to drive owners.
2. Components that could be partitioned into different communities: these graphs demonstrated high modularity.
3. Graphs consisting of usernames (logon IDs) for web applications; these graphs had high edge weights, low modularity, and a small number of nodes.
4. Graphs that were interesting but require additional study, such as component *d4c4*. While Type four graphs did display social networks, the clustering of

components, modularity, and edge weights did not mirror the communication patterns between email addresses.

Contributions of our research include a new method of analyzing relationship between email addresses, creation of a new dataset, and testing measures of centrality scores to automatically classify email addresses. We showed that our method for detecting social networks can be used to group email addresses that indicate real communication between users and those that do not. We created a new dataset by collecting images from volunteers. In addition, we extracted graphs from each drive, broke the graphs into components and recorded statistics for all graphs that demonstrated valid social networks. These datasets can be used to conduct further testing with ground truth. We also tested measures of centrality on email addresses that came from social networks and validated our results with drive owners. Our findings demonstrated that this technique can potentially be used to automatically classify email addresses.

Finally, we answer the research questions asked in Chapter 1:

Can the proximity and quantity of co-located email address pairs on secondary storage media be used to discover information about social networks?
Yes. For nine out of ten drives we were able to discover at least one social network. There were no social networks discovered on $d6$ but as previously mentioned that drive was co-located in the same laptop as. Drive $d5$ had eight social networks. The average number of social networks discovered across drives were 2.2 and the mode was 1. We discovered networks that included soccer teams, classmates, extracurricular clubs, co-workers, friends, family, and clients.

Can these properties be used to distinguish between true positives (email addresses used for communication between human users) and false positives such as text that happens to have the "@" symbol.
The proximity and the quantity of co-located email addresses did little to establish the difference between true positives and false positives. However, the components we extracted were composed of email addresses that tended to be in one category or the other. For example, all drive owners that were running Microsoft Windows had within the top four components a graph with the same Microsoft email addresses and

user-names, and for the majority of the drives the best components containing social networking data were within the top three largest components.

6.2 Future Work

While our research produced a new tool and provided another method for discovering social networks, there is room for improvement. Areas of potential improvement include reverse engineering ID strings for applications other than email, speed optimizations, community-detection, fingerprinting, resolving email address aliases, reducing noise, and social network structure hierarchy-detection. Developments in these areas would multiply the benefits for analysts conducting research or investigations.

The ability to reverse engineer ID strings would be useful in the cases where we discovered clusters of addresses with identical domain names that were written in plain-text but contained user names that were encoded with an alias such as a hash code or numbered user-ID. For Facebook user-ID's there are already Internet tools that can take a Facebook ID and provide the user-name equivalent when an ID is entered manually. This process could be automated in cases where access to online lookup tools is a feasible solution. In other cases, a deeper investigation of the ID scheme may be warranted.

Community Detection is another area that can be further studied in relation to our methods. In our research, we were able to extract on average of one to two social networks per drive. This was done by analyzing components of the drive and filtering the components by their core. However, even with our filtering techniques components representing social networks could still be very large, such as Component $d9c1$. It would be beneficial if social networks found in graphs like Component $d9c1$ could be further dissected in community structures and discoveries made about those specific communities. For example, which person is the gateway to the different community sub-sets.

In our research, some of the false positives included code that was unique to certain software libraries, email clusters that represented a network of programmers that developed a particular application, domains that were specific to certain Operating Systems including those typically installed as Virtual Machines, and clusters that represented a network of email addresses from online subscriptions that user subscribed to. These subscriptions included online magazines and newspapers. While these graphs do not depict the type of networks we are interested in, they could be used to discover what software the user is running, whether or not they have multiple Virtual Machines, and what on-line subscriptions they are receiving.

Email address alias resolution is another area for future work. Drive owners and their associates had many different email addresses, each represented as a different node in the graph. In some studies researchers may want to combine email addresses that belong to the same person. Developing a method to detect email address aliases would be helpful to clearly recognize connections between specific users instead of between specific email addresses.

In addition, it would be beneficial for researchers to develop automated methods to eliminate graphs that are not useful. Reducing noise can be accomplished by the systematic analysis of metrics related to "Useful" and "Not Useful" components. Metrics such as those given in Table 5.3 and Table 5.4 can be calculated on a large sample of "Useful" and "Not Useful" components to identify features that correlate to graphs that depict social structures. The goal is to reduce the number of graphs returned from Betographer by eliminating graphs that are not useful. As previously stated, while many of the graphs are useful for fingerprinting purposes they did not show true social networks. When searching for social networks these graphs should be ignored or eliminated.

The last suggestion is to conduct further work in centrality measurements of individual nodes to discover hierarchies in their social structure. Our results suggest that Eigenvector and Betweenness calculations can be used to give each node a score and that nodes with the highest scores tend to represent the most important people and their closest associates within the component. It would be interesting to determine whether these scores can be used to determine the hierarchical structure of the social networks and whether the convergence of centrality scores (Eigenvector, Betweenness, and Closeness) are a fundamental property that demonstrates that a component is a social network.

APPENDIX A:
Betographer Program

A.1 Betographer.py

Below is the source code for the betographer program.

```
#!/usr/bin/env python3

import networkx as nx
import sys
import os
import argparse
import timeit
import collections
from concurrent.futures import ProcessPoolExecutor, as_completed
from os.path import join, basename, normpath

def ingest_email(be_dir):
  list_dict = {}
  aliases = set()
  email_file = join(be_dir, "email.txt")

  with open(email_file, "rb") as f:
    for l in f:
      try:
        line = l.decode("utf-8")
      except:
        print("Line contains invalid utf-8; skipping")

      if line[0] == "#":
        continue
      raw_offset, raw_alias, remainder = line.split("\t", 2)

      alias  = raw_alias.replace("\\x00","")
      aliases.add(alias)

      if raw_offset.isdigit():
        root   = "top"
        offset = int(raw_offset)
      else:
        complex_offset = raw_offset.split("-")
        root   = "-".join(complex_offset[:-1])
        offset = int(complex_offset[-1])

      if root not in list_dict:
        list_dict[root] = [(offset, alias)]
      else:
        list_dict[root].append((offset,alias))

    for l in list_dict.values():
      l.sort()

  return aliases, list_dict

def nx_weighted_edge_list(edge_dict):
  w_edges = []
  for p, w in edge_dict.items():
    w_edges.append((p[0], p[1], {"weight":w}))
  return w_edges

def find_edges(feature_data, win_size=256):
  node_q = collections.deque()
  edges  = collections.defaultdict(int)

  for l in feature_data.values():
    node_q.clear()
    for offset, alias in l:
      while (node_q and (node_q[0][0] < offset - win_size)):
        node_q.popleft()

      for n in node_q:
        pair = tuple(sorted([alias, n[1]]))
        if pair[0] != pair[1]:
          edges[pair] += 1

      node_q.append((offset, alias))

  weighted_edges = nx_weighted_edge_list(edges)

  return edges, weighted_edges

def make_gephi(V, E):
  G = nx.Graph()
  G.add_nodes_from(V)
  G.add_edges_from(E)
  return G

def make_weight_dict(edge_weights):
  w_dict = {}
  for pair, weight in edge_weights.items():
    if weight not in w_dict:
      w_dict[weight] = [pair]
    else:
      w_dict[weight].append((pair))
  return w_dict

def find_big_connected_components(G, num=5):
  n = nx.number_of_nodes(G)
  con_comps  = list(sorted(nx.connected_component_subgraphs(G), key=len, reverse=True))
  return con_comps[:num]
```

```
class Summary:
  def __init__(self, path):
    self.lines = []
    self.path  = path

  def print_last_n(self, n=0):
    index = 0 if n == 0 else len(self.lines) - n
    for l in self.lines[index:]: print(l)

  def write(self):
    with open(self.path, "w") as f:
      for l in self.lines: print(l, file=f)

  def add(self, line):
    self.lines.append(line)

def get_dirs(dir_list):
  dirs = []
  with open(dir_list, "r") as d:
    for line in d:
      dirs.append(line.strip().rstrip(os.sep))
  return dirs

def process_many(in_dirs, num_jobs, outdir, args):
  out_dirs  = [join(outdir, basename(subdir)) for subdir in in_dirs]
  arguments = zip(in_dirs, out_dirs, [args]*num_jobs, [True]*num_jobs)
  with ProcessPoolExecutor() as executor:
    executor.map(process_one, arguments)

def process_one(arguments):
  be_dir, out_dir, args, multi_mode = arguments
  bar      = "="*80
  win_size = args.win_size
  no_loner = args.drop_loners
```

```
big_subs = args.big_subgraphs
devel    = args.developer
gname    = basename(normpath(be_dir)) + "_" + str(win_size) + ".gexf"
sumfile  = basename(normpath(be_dir)) + "_" + str(win_size) + "_summary.txt"

if not os.path.exists(out_dir):
  os.makedirs(out_dir)

sumpath  = join(out_dir, sumfile)
gpath    = join(out_dir, gname)
summary  = Summary(sumpath)
summary.add("BEtographer summary for command line invocation:")

summary.add(" ".join(sys.argv))
summary.add(bar)
summary.add("Using Window Size: {}".format(win_size))
summary.add("Writing graphs to: {}".format(gpath))
summary.add(bar)
if not multi_mode: summary.print_last_n()

start             = timeit.default_timer()
nodes, email_data = ingest_email(be_dir)
ingest_time       = timeit.default_timer()

summary.add("Ingested in {} seconds".format(ingest_time - start))
if not multi_mode: summary.print_last_n(1)

edges, weighted_edges = find_edges(email_data, win_size)
edge_time             = timeit.default_timer()

summary.add("All neighbors linked in {} seconds".format(edge_time - ingest_time))
if not multi_mode: summary.print_last_n(1)
if no_loner:
  friends = set()
  for e in edges.keys():
    friends.add(e[0])
    friends.add(e[1])
  nodes = friends

G = make_gephi(nodes,weighted_edges)
nx.write_gexf(G, gpath)
end = timeit.default_timer()

summary.add("Finished in {} seconds".format(end - start))
if not multi_mode: summary.print_last_n(1)
summary.add(bar)
summary.add("Summary Statistics for Whole Graph:")
summary.add(bar)
summary.add(nx.info(G))

if big_subs:
  subgraph_start = timeit.default_timer()
  i = 1
  summary.add("Looking for the top {} subgraphs".format(big_subs))
  if not multi_mode: summary.print_last_n(1)

  major_subs = find_big_connected_components(G, big_subs)
  subgraph_end = timeit.default_timer()
  subgraph_time = subgraph_end - subgraph_start

  summary.add("Found {} big subgraphs in {} seconds.".format(len(major_subs),
                                                              subgraph_time))
  if not multi_mode: summary.print_last_n(1)

  for g in major_subs:
    summary.add(bar)
    summary.add("Summary Statistics for subgraph {}:".format(i))
    summary.add(bar)
    summary.add(nx.info(g))
    gname = basename(normpath(be_dir)) + "_" + str(win_size) + "sub" + str(i) + ".gexf"
```

```
      out   = join(out_dir, gname)
      nx.write_gexf(g, out)
      i += 1

  if devel:
    experimental(G, edges)

  summary.write()

  if multi_mode:
    print(".", end="")
    sys.stdout.flush()
  else:
    print("Check {} for a summary of the results.".format(sumpath))

if __name__=="__main__":
  parser = argparse.ArgumentParser(description='Discover networks in bulk'
                                               ' extractor email.txt output.')

  group = parser.add_mutually_exclusive_group(required=True)

  group.add_argument('be_results_path', nargs='?',
                    help="Path to bulk_extractor output files. Required unless"
                         " running with the -p option.")

  parser.add_argument('-o', '--output-path', required=True,
                     help="Sets the directory where betographer should store its"
                          " output (required).")

  parser.add_argument('-w', '--win-size', type=int, default=256,
                     help='Optionally sets the window size in bytes. Default is 256.')

  parser.add_argument('-d', '--drop-loners', action='store_true',
                     help='Drop all nodes with no edges to other nodes.')
```

```
parser.add_argument('-m', '--big-subgraphs', metavar='N', type=int,
                    help='Save a separate gexf file for the top N largest'
                         ' connected components of the graph.')

group.add_argument('-p', '--parallel', dest="dir_list",
                    help='Specify name of a file containing a list of bulk'
                         ' extractor output directories and process these'
                         ' in parallel. An output directory will be created'
                         ' for each as a subdirectory of the output path.')


args     = parser.parse_args()

if args.dir_list:
  multi_start = timeit.default_timer()
  in_dirs   = get_dirs(args.dir_list)
  num_jobs  = len(in_dirs)
  process_many(in_dirs, num_jobs, args.output_path, args)
  very_end = timeit.default_timer()
  print()
  print("Completed {} jobs in {} seconds.".format(num_jobs, very_end - multi_start))
else:
  process_one((args.be_results_path, args.output_path, args, False))
```

APPENDIX B:
Node and Edge Counts

B.1 Node and Edge Counts

Table B.1: A Count of the Nodes and Edges in Each Graph.

	1		2		3		4		5		6		7		8		9		10	
Sub-graphs	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges	Nodes	Edges
Graph 1	9324	24751	9035	23617	24075	64624	16214	31634	10	24	28658	365054	7655	33789	12283	33968	534	819	19953	62610
Graph 2	6087	26847	867	1769	2469	6577	10192	39406	8	7	599	598	4640	11672	1630	2287	9	21	282	1266
Graph 3	270	295	277	363	260	667	692	821	6	5	297	312	277	363	495	835	8	7	277	363
Graph 4	104	138	188	670	249	259	328	1263	6	5	277	363	233	657	91	105	8	14	129	146
Graph 5	73	72	188	191	168	546	148	202	6	5	187	186	118	290	79	97	5	10	94	161
Graph 6	69	80	75	131	100	286	141	261	5	4	132	131	63	181	70	205	5	5	93	115
Graph 7	61	78	41	51	100	192	123	167	4	3	62	81	59	153	65	115	5	4	80	168
Graph 8	58	74	35	99	75	157	110	267	4	4	51	90	41	69	56	97	5	8	78	95
Graph 9	55	81	31	35	57	73	107	136	3	2	38	80	37	135	54	105	4	4	70	93
Graph 10	55	76	30	83	51	99	61	116	3	2	36	148	36	90	39	73	4	3	70	170
Graph 11	52	53	30	31	48	87	58	84	3	3	32	37	30	36	36	56	4	4	67	67
Graph 12	51	50	28	66	46	89	57	57	3	3	26	25	26	42	35	107	4	3	54	63
Graph 13	47	77	28	74	45	149	54	58	3	3	23	82	25	25	35	99	3	3	49	137
Graph 14	46	56	25	31	45	95	53	53	3	3	23	61	24	70	34	37	3	2	48	106
Graph 15	43	70	24	48	44	85	51	90	2	1	23	88	23	41	33	55	3	3	48	89
Graph 16	42	43	23	56	42	74	44	60	2	1	23	52	23	37	31	76	3	2	46	145
Graph 17	41	122	22	25	39	61	43	42	2	1	20	25	22	26	30	34	3	2	44	119
Graph 18	36	38	21	22	39	75	40	50	2	1	20	42	21	50	28	66	3	2	44	72
Graph 19	35	99	21	34	38	69	38	40	2	1	18	39	19	39	27	37	3	3	36	68
Graph 20	35	66	20	25	38	37	36	47	2	1	18	62	19	58	25	26	3	3	36	87

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C:
Exit-Interview Questions for Drive Owners

C.1 Exit-Interview Questions for Drive Owners

The following are questions which were asked during exit-interviews with those who volunteered their storage media:

1. You will be presented with a list of email addresses. For each email address, indicate whether you recognize the email address.

2. If you recognize an email address, please provide the following additional information:

- Is this an address of a person who has (to your knowledge) used your computer?
- Is this an address of a person with whom you communicate by email?

3. In general, does the graph accurately describe the relationship between the correspondents listed?

4. If the graph is clustered into different groups, are you able to discern a reason for the clustering (for example, different groups may represent coworkers, family, friends, etc.)?

5. What operating system do you use (e.g. Windows 7, Windows 8, OSX 10.7, etc.)?

6. What email client do you use (Outlook, Outlook Express, Apple Mail, Web-based email, other (please describe))? If you use multiple clients (for example if you use Outlook for official email and Gmail for personal email), please list all of them and approximate the percent of time used on each.

7. How long have you used this computer?

8. Have there been any prior users / owners of this computer?

9. Do you share the computer with anyone else?

10. How often do you communicate by email (daily / weekly / monthly)?

11. Why do you believe that these email addresses were collected as pairs? Are these emails frequently on the same address lines for emails or some other situation such as used for log on purposes or all located within the same webpage? If so how often are they used or accessed for this purpose?

12. The clusters shown on the graph are assumed to be related. To your knowledge is that true? Do the clusters represent a group such as a group of co-workers or family members?

13. Are you the only user of this hard drive? If not how many people and how often do they use it? Did this hard drive have a prior owner? How long have you been using the hard drive?

14. Is there anything else you would like to add? For example, if the analysis you were presented with is incorrect, do you have an explanation about what went wrong?

List of References

[1] "Bulk extractor 1.4," 2015. [Online]. Available: http://digitalcorpora.org/downloads/bulk_extractor/BEUsersManual.pdf

[2] B. Bollobás, *Modern Graph Theory*. New York, New York: Springer Science & Business Media, 1998, vol. 184.

[3] D. Knoke and S. Yang, *Social Network Analysis*. Thousand Oaks, California: Sage Publications, 2008.

[4] A. D. R. et. al. (2011). Social network analysis: Theory and application. [Online]. Available: http://train.ed.psu.edu/WFED-543/SocNet_TheoryApp.pdf

[5] M. Kilduff and W. Tsai, *Social Networks and Organizations*. Thousand Oaks, California: Sage Publications, 2003.

[6] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks." *Nature: International Weekly Journal of Science*, vol. 393, no. 6684, pp. 440–442, 1998.

[7] L. C. Freeman, "Centrality in social networks: Conceptual clarification," *Social Networks*, vol. 1, pp. 215–239, 1978.

[8] P. Carringon et al., *Gephi: Features*. New York, NY: Cambridge University Press, 2005.

[9] D. Krackhardt, "Graph theoretical dimensions of informal organizations," *Computational Organization Theory*, vol. 89, no. 112, pp. 123–140, 1994.

[10] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 1, pp. 36–41, Dec. 2006.

[11] S. B. Seidman, "Network structure and minimum degree," *Social Networks*, vol. 5, pp. 269–287, 1983.

[12] S. Garfinkel, "Digital forensics," *American Scientist*, vol. 11, p. 370, Sep. 2013.

[13] S. L. Garfinkel, "Digital forensics research: The next 10 years," *Digital Investigation*, vol. 7, 2010.

[14] S. L. Garfinkel, "Forensic feature extraction and cross-drive analysis," *Digital Investigations*, vol. 3, pp. 71–81, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1016/j.diin.2006.06.007

[15] R. Marty, “Cyber security: How visual analytics unlock insight,” in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining)*, Chicago, Illinois, Aug. 2013, pp. 1139–1139.

[16] S. Teerlink and R. Erbacher, “Improving the computer forensic analysis process through visualization,” *AMC*, vol. 49, no. 2, pp. 71–75, Feb. 2006.

[17] B. Scneidermann, “Tree visualization with tree-maps: 2-d space filling approach,” *ACM*, vol. 11, no. 1, pp. 92–99, Aug. 1992.

[18] N. Rowe, “Identifying forensically unintersting files using a large corpus,” in *5th International Conference on Digital Forensics and Computer Crime*, Moscow, Russia, Sep. 2013, pp. 1–11.

[19] W. Campbell et al., “Social network analysis with content and graphs,” *Lincoln Labs*, vol. 20, pp. 62–81, Sep. 2013.

[20] C. Bird et al., “Minning email social networks,” in *2006 International workshop on Mining software repositories*, Shanghai, China, May 2006, pp. 137–143.

[21] S. Boucher and B. Kuange. (2011, May). Email evidence-now you see it, now you don’t. [Online]. Available: http://www.forensicfocus.com/email-evidence-now-you-see-it. Accessed May 9, 2015.

[22] N. Rowe et al., “Making sense of email addresses on drives,” *Journal of Digital Forensics, Security and Law*, vol. 9, no. 2, 2016.

[23] S. Garfinkel et al., “Bringing science to digital forensics with standardized forensic corpora,” *Digital Investigations*, vol. 6, pp. S2–S11, 2009.

[24] “Index of be-stoplists.zip,” June 15,2015. [Online]. Available: http://digitalcorpora.org/downloads/bulk_extractor/

[25] T. M. Fruchterman and E. M. Reingold, “Graph drawing by force-directed placement,” *Software: Practice and experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[26] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, “Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software,” *PloS one*, vol. 9, no. 6, p. e98679, 2014.

[27] B. Slawski. (2006). Google’s most popular and least popular top level domains. [Online]. Available: www.seobythesea.com/2006/01/googles-most-popular-and-least-popular-top-level-domains/

[28] K. Hampton et al. (2011). Part 3: Social networking site users have more friends and more close friends. [Online]. Available: www.pewinternet.org/2011/06/16/part-3-social-netowrking-site-users-have-more-friends-and-more-close-friends/

[29] A. Smith. (2014). 6 new facts about Facebook. [Online]. Available: www.pewresearch.org/fact-tank/2014/02/03/6-new-facts-about-facebook/

[30] N. Warburton. (2013). Robin Dunbar on Dunbar numbers. [Online]. Available: www.socialsciencespace.com/2013/11/robin-dunbar-on-dunbar-numbers/

[31] K. Woods et al. (2011). Creating realistic corpora for security and forensic education. [Online]. Available: http://simson.net/clips/academic/2011.ADFSL.Corpora.pdf

[32] "Social network analysis: Theory and application," 2016. [Online]. Available: https://gephi.org/features/

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California